DCA952-803-42(a)
ETNAM SD VOL III REVISION 1

DEFENSE COMMUNICATIONS AGENCY

NMCS TECHNICAL SUPPORT DIRECTORATE

SYSTEM DESCRIPTION

EUROPEAN THEATER NETWORK ANALYSIS MODEL

(ETNAM)

VOLUME III

ANALYTICAL MANUAL

JANUARY 1971

PREPARED FOR LOGISTICS DIRECTORATE (J-4)

UNCLASSIFIED

Security Classification

DOCUMENT CONTROL DATA - R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author)	2a. REPORT SECURITY CLASSIFICATION
Research Analysis Corporation McLean, Virginia 22101	Unclassified
	2b. GROUP N/A

3. REPORT TITLE

SYSTEM DESCRIPTION
VOLUME III - ANALYTICAL MANUAL
EUROPEAN THEATER NETWORK ANALYSIS MODEL (ETNAM)

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

Final Report

5. AUTHOR(S) *(First name, middle initial, last name)*

John E. Cremeans, Henry S. Weigel, Dorothy L. Ray, and John T. Sincavage

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
January 1971	190	4

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
DCA 100-70-C-0039	DCA 952-803-42(a)
b. PROJECT NO.	
c.	9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*
d.	

10. DISTRIBUTION STATEMENT

Approved for Public Release; Distribution Unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	Defense Communications Agency NMCS Technical Support Directorate Washington, D. C. 20305

13. ABSTRACT

1. This document provides a basic, non-mathematical introduction to the ETNAM model and the associated system of computer programs. It provides a basic guide to the preparation of input data and the operation of the system of computer programs from an analytical point of view. It is intended to serve the military analyst as a basic introduction and guide to the application of the ETNAM model to mobility problems. ETNAM is an optimizing model that uses a special form of the simplex algorithm of linear programming and the shortest chain algorithm of graph theory to select routes and allocate resources to those routes so as to minimize cost, minimize time or maximize flow subject to the constraints of the network and the available resources.

2. This document supersedes the Research Analysis Corporation's publication, "System Description, Part II - Analytical Manual, dated October 1969.

DD FORM 1 NOV 68 1473

UNCLASSIFIED

Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
intra-theater mobility						
linear programming						
mathematical programming						
model, transportation network						
multicommodity						
network analysis						
problems, theater mobility						
strategic mobility						
theater mobility						
transportation network model						

DCA 952-803-42(a)
ETNAM SD Vol III, Rev 1

DEFENSE COMMUNICATIONS AGENCY

NMCS TECHNICAL SUPPORT DIRECTORATE

WASHINGTON, D. C. 20305

SYSTEM DESCRIPTION

VOLUME III — ANALYTICAL MANUAL

EUROPEAN THEATER NETWORK ANALYSIS MODEL (ETNAM)

PREPARED FOR LOGISTICS DIRECTORATE (J-4)

The Research Analysis Corporation has provided assistance in the preparation of this document in performance of technical support to DCA under Contract 100-70-C-0039.

JANUARY 1971

EUROPEAN THEATER NETWORK ANALYSIS MODEL (ETNAM)

SYSTEM DESCRIPTION - VOLUME III

ANALYTICAL MANUAL

FOR THE

LOGISTICS DIRECTORATE (J-4), JOINT STAFF

ABSTRACT:

1. This document provides a basic, non-mathematical introduction to the ETNAM model and the associated system of computer programs. It provides a basic guide to the preparation of input data and the operation of the system of computer programs from an analytical point of view. It is intended to serve the military analyst as a basic introduction and guide to the application of the ETNAM model to mobility problems. ETNAM is an optimizing model that uses a special form of the simplex algorithm of linear programming and the shortest chain algorithm of graph theory to select routes and allocate resources to those routes so as to minimize cost, minimize time or maximize flow subject to the constraints of the network and the available resources.

2. This document supersedes the Research Analysis Corporation's publication, "System Description, Part II - Analytical Manual, dated October 1969.

CONTENTS

ILLUSTRATIONS

TABLES

CHAPTER 1. INTRODUCTION

1. General - The System Description. The System Description (SD) for the European Theater Network Analysis Model (ETNAM) has been prepared for the Defense Communications Agency (DCA) under contract DCA 100-70-C-0039, Modification #G01, dated January 22, 1970. Submission of this document is in fulfillment of paragraph 3.2 of the contract.

a. The System Description consists of four related volumes bound separately. Volume I is the User's Manual and is intended primarily for use by the action officers. Volume II is the Operator's Manual and was written by the National Military Command System Support Center (NMCSSC) and is intended for the use of computer operators and others interested in the operating characteristics of the model. Volume III, this document, is the Analytical Manual and is intended for the use of analysts and programmers. Volume IV is the ETNAM Data Base, consisting of two parts, and is intended for users and analysts as a source of network and mobility resource data.

b. While it is intended that these manuals be capable of use independently, it should be noted that the Operational Capability Description (OCD) and the Capability Design Specification (CDS) for ETNAM contain additional information of value to both the action officers and analysts. The OCD contains a detailed statement of the types of applications for which the ETNAM model has been designed. The CDS (particularly Appendices A and B) contains a detailed and rigorous mathematical statement of the algorithms used in the ETNAM model. Appendices A and B of the CDS are repeated in this manual as Enclosures 1 and 2.

c. The System Description has been prepared in accordance with the DCA Instruction 210-175-3 "Documentation Standards" for this type document. The order of the major headings and the information selected for inclusion have been based upon this instruction. The User's Manual (UM) follows the suggested outline exactly. The suggested outline for this Analytical Manual (AM) has been modified slightly to permit the presentation of all the information about each of the four modules of the system in separate chapters.

d. This Analytical Manual is intended to provide a basic, non-mathematical introduction to the ETNAM model and the ETNAM system of computer programs. It provides a basic guide to the preparation of input data and the operation of the computer program from an analytical point of view. It discusses the methodology used, the adequacy of input data for European Theater strategic mobility problems, and alternative solutions available. Each of the four modules of the computer program is discussed in some detail. The analyst should be able to use this information to obtain a basic understanding of the model and of the required input data and its format.

e. It should be noted at the outset that the System Description will not be adequate in itself to prepare the user for the operation of the ETNAM model. The formulation of strategic mobility problems for ETNAM requires some basic familiarity with network terms and methods. Understanding of the solution algorithm of the model requires an understanding of some sophisticated mathematical techniques.

f. The basic formulation of the problem to be solved and the interpretation of the results should be carefully controlled by the user himself. In the applications of the ETNAM model completed to date, those with full user participation have been significantly more successful than those conducted entirely by the model developers.

g. It has also been discovered that for many users the quickest and most successful way to learn the uses of the ETNAM model is to study the formulation and results of previous applications. A number of these test cases are now available at the Research Analysis Corporation and in the Systems Analysis Division of the Office of the Deputy Director for Logistics Strategic Mobility, J-4, JCS.

2. Problem-Process/Event Simulation. The definition of the problem to be solved by ETNAM was developed after a study of users' requirements. A feasibility study was conducted to determine the best method available for use in solving this problem and a "breadboard" version of the present ETNAM model was developed and tested. These developments and others leading to the present ETNAM model are described in detail in the Feasibility Study, the Operational Capability Description (OCD), the Capability Design Specification (CDS) and the Verification Test Plan (VTP). Those analysts interested in more details of the development of the system should refer to these documents.

a. Operating Objectives. In general terms the ETNAM model was designed to provide a tool for the detailed analysis of the capabilities of a transportation system and to determine the optimum allocation of resources in that system. The model has four objectives of operation.

(1) In the maximum-flow (max-flow) objective, the model will determine the maximum flow of men and materials that can be sustained by a transportation network given the origin-destination pairs, the inventory of mobility resources available, the capacities of the road, rail, air, waterway, pipeline and other types of links in the transportation network, and the capacities of specified nodes of intersection of three or more links.

(2) In the minimum-cost (min-cost) objective, the model will determine the set of routes and the allocation of resources to those routes that will minimize cost, given the delivery requirements, the maximum number of each type of mobility resource that may be assigned, the relative costs of each of these resources, and the capacities of each of the links and specified nodes of the transportation network.

(3) In the minimum-time (min-time) objective, the model will determine the set of routes and the allocation of resources to those routes that will minimize flow unit-hours, given the delivery requirements, the maximum number of mobility resources that may be assigned, the relative speeds of each of the modes, and the capacities of each of the links and specified nodes of the transportation network. The flow units may be all in tons if all the commodity units have been converted to equivalent tons, or they may be in the commodity's natural unit. This option will be explained later in this section.

(4) Finally, there is a mixed minimization of cost and time objective in which the model will determine the set of routes and the allocation of resources to those routes which will minimize the combination of both cost and time according to weights established by the user.

(5) In summary, the model may be used in the max-flow, min-cost, min-time or mixed minimization of cost and time operating objectives.

b. Max-Flow. The max-flow objective may be called a capabilities objective. That is, when operated in this objective the model determines the maximum capability of the transportation system, given the limitations of the available resources and the capacity of the network. It should be emphasized, however, that this maximum flow is strictly dependent upon the origin-destination pairs provided as input data. As a simple example, consider a problem involving only two origin-destination pairs: one pair fifty airline miles apart and the second one hundred airline miles apart. If resources are limited, the model will assign all available resources to the first pair and possibly none to the second pair because this allocation will provide the maximum flow. Thus the maximum-flow operational objective of the model will not be useful unless the meaning of flow maximization is understood. In general, the max-flow objective is not used except when a general order-of-magnitude estimate of a network capability is desired.

c. Min-Cost. The minimum-cost objective is most often used because it provides an estimate of requirements for meeting a specific delivery plan. In this operating objective the model determines the minimum-cost set of routes and mobility resources necessary to fulfill delivery requirements. It may be used to determine the feasibility of a movement plan or to determine the mobility resource requirements to meet delivery goals.

d. Min-Time. The minimum-time operating objective determines the mobility resource requirements and the routes that will move men and materials through the network to their destination in the minimum time. It is important to understand that the model minimizes flow unit-hours. That is, it minimizes the sum of the products of all deliveries and the elapsed time from origin to destination. Users may wish to examine both the min-cost and the min-time solutions to the same basic problem.

In general, a comparison of a min-time solution with a min-cost solution will show that men and materials to be moved have been shifted from slower (but lower cost) modes such as waterway to faster (but higher cost) modes such as air.

e. Mixed Minimization of Cost and Time. The mixed minimization of cost and time objective is not a separate operating objective as far as ETNAM is concerned, but rather a problem formulation option that the analyst may use under the min-cost operating objective. Briefly, the method assigns time as a resource to be minimized along with the other applicable costs. For each commodity - transportation mode pair a time "productivity" is given. This is the amount of time it takes to move one unit of the specified commodity one mile on the specified transportation mode. A price is attached to the resource time. In a min-cost run, time is minimized to the extent that the total cost is minimized which depends on the prices of the other resources. It is not necessary to follow this procedure for all commodities. It is frequently the case that some commodities should be given time-priorities while others should be shipped via the least cost routes. For example, PAX and Unit Equipment might have elapsed time included as one of the resources used (and priced) while other commodities are handled in the usual min-time way. This approach would have the effect of placing PAX and Unit Equipment shipments on the most expeditious routes while the other commodities (e.g., POL) would be sent via the least cost routes without the time minimization consideration.

f. Problem Formulation. A problem for ETNAM may be formulated in either of two ways. One way is to convert all flows and requirements for commodity movement into some equivalent unit such as short tons. This allows the network capacities for the links and nodes to be given in short tons per day. Another way is to express all commodities and commodity requirements in terms of their natural units (e.g., numbers of passengers, barrels of gasoline, tons of coal, etc.). This conveniently allows the expression of network capacities in terms of vehicles-per-time-period, which is the way in which most traffic statistics are collected. The "equivalent unit" formulation was the basis for the original ETNAM development and is described in more detail in Enclosures 1 and 2. The "natural unit" formulation has subsequently been incorporated in the ETNAM system and is fully described in Enclosure 3.

g. Outputs. The outputs of ETNAM provide considerable detail about the transportation system under study. The solution itself is presented in several forms. The routes chosen and the resources assigned to them are provided by commodity. The movements by commodity over each of the links of the network are presented. If the network capacities are expressed in some equivalent unit, like short tons, a unit-mile or ton-mile flow by mode is given. The total flow through origin and through destination nodes is shown. Total flow through up to fifty intermediate nodes may be accumulated at the user's option. Summaries of commodity flow and of resource utilization are also presented. Analytical data

is presented that indicates the decrease in cost (or the increase in flow) that could be obtained if one or more units of each mobility resource were available. The decrease in cost (or the increase in flow) that would result from one more ton of capacity on each of the links or constrained nodes in the network is also presented. In linear programming these analytical data are referred to by several names including shadow prices, marginal costs, and dual solution.

3. Discussion of Methodology. The ETNAM System consists of an Input Module, a RACAT Module, an Output Module, and a POP (Post Optimal Package) Module. Figure 1 shows them schematically. These modules are discussed in detail in Chapters 2, 3, 4 and 5 respectively. The ETNAM system is based upon a generalized algorithm called the Resource Allocation and Chain Analysis Technique (RACAT) that was developed specifically for this purpose. The Input and Output Modules are primarily data processing modules designed solely to edit and format the inputs and outputs for the RACAT Module. The POP Module gives upper and lower limits to which the right-hand-sides (i.e., link capacities, resource inventories, node capacities and requirements) may be changed, one at a time, without requiring a change in the current solution or basis. If this information is desired then POP is employed after a solution from RACAT is obtained. POP is fully described in Enclosure 4.

a. Input Module. All inputs to the ETNAM system are prepared manually. That is, there is no automated system for the preparation of the input data involving the selection of existing data from the National Military Command and Control System (NMCCS) data banks. A typical run of the ETNAM system requires less than one thousand punched cards. Of these, approximately two thirds are network data cards that are based upon theater data and hence apply to all problems for that theater. The Input Module is thus a relatively simple computer program that edits and formats the input data and prepares reports that can be checked by the user. Complete descriptions of these reports are contained in the User's Manual.

b. Output Module. This module takes the output of the RACAT Module and edits and format it for presentation to the user. The raw output of the RACAT Module is the mathematical solution to the problem and is not easily understood or interpreted. Some special problems must be solved in organizing and editing this data for presentation, but standard data processing techniques are used. In general, the Output Module reports have been found to be useful and complete in the applications processed to date. These reports are also described in the User's Manual.

c. RACAT Module. This module contains the mathematical algorithm and is the core of the system. It is an extension to the multi-commodity network flow algorithm suggested by Ford and Fulkerson in 1958 and

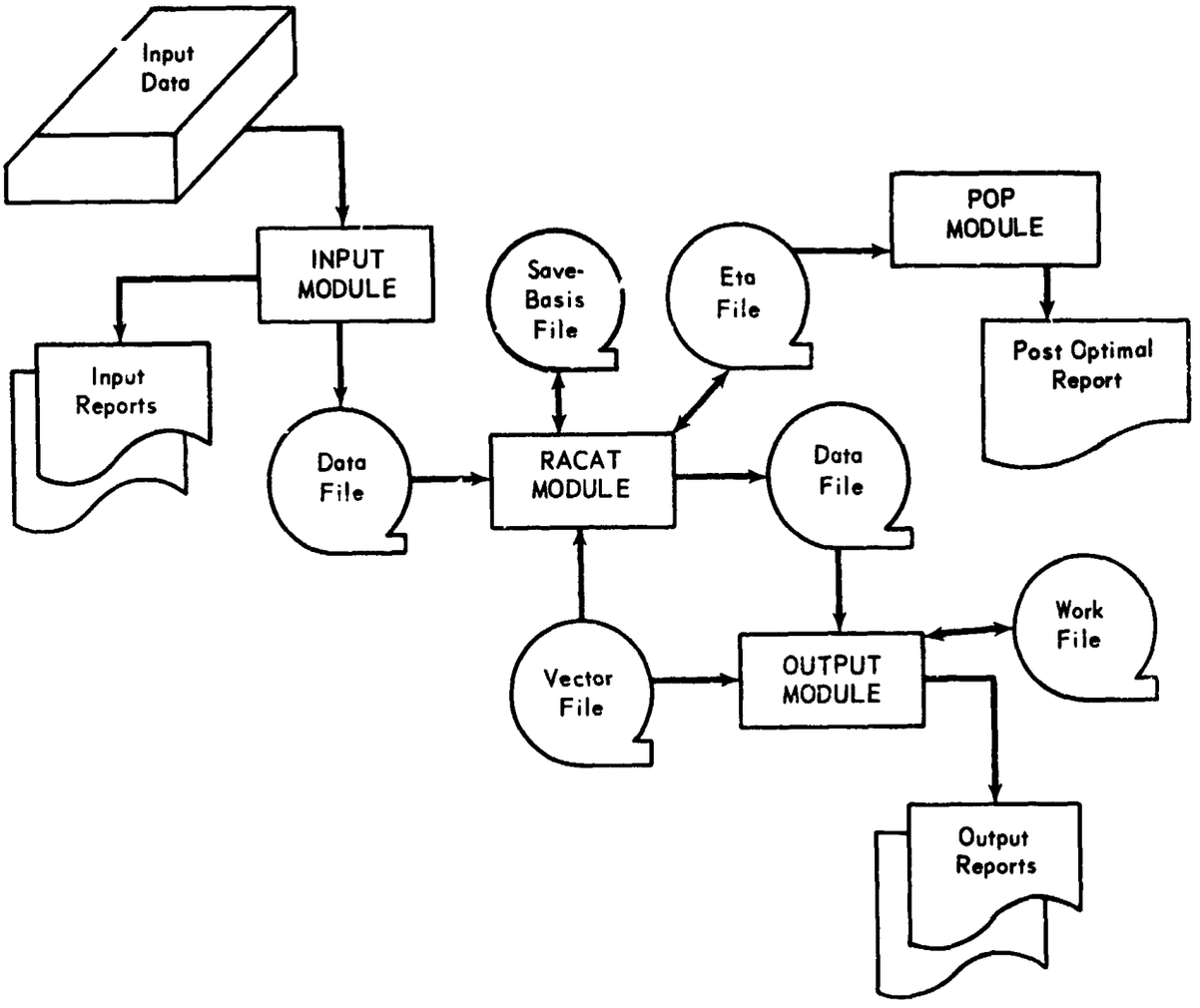


FIGURE 1. ETNAM SYSTEM FLOW SCHEMATIC

extended to the minimum-cost formulation by Tomlin in 1966[1]. This algorithm finds the optimum solution to network flow problems where there are distinct origin-destination pairs to be connected by routes in such a way that all requirements are satisfied and none of the individual link and node capacities nor the resource inventories are exceeded. There are other models that will find solutions to the problem, but these are not optimal solutions and there is no way to determine how far these solutions are from the optimum.

(1) A simple example of a multi-commodity problem is the problem of the telephone company on Mother's Day. Thousands of sons are calling thousands of mothers and the telephone company must connect each son to the corresponding mother in such a way that none of their telephone lines is overloaded. There are excellent single-commodity models that will find the least cost solution to single-commodity problems. If the single-commodity model were used to solve the telephone company's problem outlined above, it would in effect connect each son to the nearest mother -- an unsatisfactory solution for most sons and mothers.

(2) A number of multi-commodity computer models have been written using the basic Ford and Fulkerson method. The Logistics Research Project of the George Washington University, the Civil Engineering Department of the Johns Hopkins University, the University of California at Berkley, and others have such models. The RACAT algorithm is different from these in that it includes node capacity and resource constraints as well as link capacity constraints.

(3) Mobility resources are likely to be scarce in most military theaters. In the European Theater the transportation network itself is one of the most dense and most sophisticated in the world. The capacities of the roads, railroads, waterways, pipelines, and airways -- while they may constrain individual movements -- are large, and there are sufficient alternatives that almost any delivery requirements can be met. The problem is to simultaneously select the routes and allocate resources to those routes in an optimal fashion. The RACAT algorithm is an optimizing algorithm that simultaneously selects routes and allocates resources while maximizing flow, minimizing cost, or minimizing time.

(4) The RACAT algorithm (and the original Ford-Fulkerson algorithm) is a combination of the simplex algorithm of linear programming and the shortest chain algorithm of graph theory. In highly simplified terms the simplex algorithm can be said to consist of 4 steps as follows:

---

[1] L. R. Ford and D. R. Fulkerson, "A Suggested Computation for Maximal Multi-Commodity Network Flows", Mgmt. Sc., October, 1958; J. A. Tomlin, "Minimum-Cost Multicommodity Network Flows", ORSA, December, 1966.

(a) Find a feasible solution consisting of a set of routes and the corresponding resources that will meet the requirements.

(b) Select a route not now used that will improve that solution.

(c) Update the solution. This may mean dropping a route in the previous solution.

(d) Continue repeating steps (b) and (c) until there is no available route that will improve the solution. When there is no such route, an optimum solution has been found.

(5) In the RACAT version of the simplex algorithm, the shortest chain algorithm is used to find the improving routes. That is, the shortest chain algorithm is substituted for step (b).

(6) In order to fully understand the solution procedure and the computer code of the RACAT Module, it is necessary to have a background knowledge of the simplex algorithm and certain aspects of graph theory. While this background knowledge is not necessary to use the model or understand its results, such knowledge will be necessary to anyone who plans to maintain or modify the computer program. A bibliography of books and articles on the subject is included as Enclosure 6. Enclosures 1, 2 and 3 are technical papers on the RACAT algorithm and contain a rigorous mathematical treatment of the algorithm. Analysts who wish to understand the procedure in all its detail should refer to these technical papers. Sample data listings appear in Enclosure 5.

d. POP Module. This module is a post optimal procedure giving the analyst data sensitivity information. The sensitivity analysis that is done is right-hand-side-ranging. The right-hand-sides referred to are the constraint limits of the problem, that is, network capacity and resource inventory constraints and movement requirements. During a POP computer run any number of right-hand-sides may be analyzed one at a time. That is, while the "range" of a given right-hand-side is found, the others are held fixed.

(1) The range of a right-hand-side is the set of values the right-hand-side may assume, keeping the other right-hand-side values fixed, without requiring a change in the solution. This information may give insight to the analyst regarding binding or potentially binding constraints for that solution. It must be remembered that this is a local sensitivity analysis, that is, a change in the problem could give quite different sensitivity results.

(2) Although POP makes use of the properties of the simplex procedure, it is not necessary to understand this procedure to be able to read the output reports. However, one should be aquainted with the mathematical techniques used to understand the computer code. Technical

information may be found in the linear programming references in the bibliography in Enclosure 6. Enclosure 4 gives a rigorous mathematical treatment of the solution procedure employed by POP.

4. Adequacy of Available Data.

a. In general, three types of data are needed to formulate a problem for the ETNAM system. These are network data, resource data, and control parameters.

(1) The network data tend to be fixed for all problems involving a given theater. New highway and railway links are constructed and old ones expanded, but these changes are infrequent. Thus, the same set of network punched cards can normally be used to run a whole series of runs in a given theater.

(2) Basic resource data change very little. New vehicles and transportation techniques are developed, but these changes are infrequent. Nevertheless resource productivities and resource prices (ten-year systems costs) are normally developed carefully for each run or series of runs. The reason is that resource productivities are a function of the operating procedures that are planned for the vehicles as well as their physical characteristics. The productivity of a carrier will be quite different if a twenty-hour operating day is planned than if an eighteen-hour operating day is planned. The relative prices to be used will also vary from study to study depending upon the ground rules agreed upon. It is anticipated, therefore, that resource data will be updated for each study.

(3) The control parameters change from run to run even within the same study and must, therefore, be prepared each time. The control parameters specify the operating objective of the individual run, the number of iterations between inversion and other parameters having to do with the control of the program during the solution of the problem. These specifications are reasonably straight-forward and the development of this data is not difficult.

b. There are three major sources of data of the first two types that have been regularly used to date. The Defense Intelligence Agency (DIA) has available extensive network data for many theaters throughout the world. Some users prefer to make their own estimates of capacity based on DIA base data. Additional information on the European Theater is available in the DCA report (prepared by RAC) on the European Theater Transportation System (ETTS). Volume IV of the SD, the ETNAM Data Base (also prepared by RAC), contains related information on Europe, Southeast Asia and Northeast Asia.

c. The Research Analysis Corporation has on file a European network in proper punched card form. These network data are based on DIA data modified as a result of the ETTS report. It has been used for a number of applications.

d. In general terms, the available data for the operation of the ETNAM model has been found to be adequate for the European Theater. This does not mean that there is not room for improvement. The ETNAM model is very sensitive to resource productivities and to resource prices. Very small changes in either of these categories of data may cause significant changes in the solution. For this reason, systems costs and productivities should be carefully reviewed by the user.

e. Based on experience in the application of the ETNAM model to date, it is strongly recommended that the user review and study resource prices and productivities prior to the first run of the model. Such a review insures that the user is satisfied with these data and, in addition, seems to assist the user in understanding the basic approach of the ETNAM model. While there is no doubt that there are adequate data available for the operation of the system, only the user can determine if these data are satisfactory for his purposes.

5. Alternative Solutions. There are two senses in which alternative solutions to problems structured for the ETNAM model may be considered. There are alternative solutions in a mathematical sense and there are alternative solutions in an operational sense.

a. The solution generated by the ETNAM model is an optimum solution to the network flow problem presented to it. Mathematically there are only two possible cases: one, there is one and only one optimum solution; and two, there is an infinite number of optimum solutions. In operational terms, the problem may have only one set of routes and one set of flows on those routes that is optimum. That is, all other possible sets cost more (or take more time, or achieve less flow).

b. Maximum flow problems frequently have alternative solutions in the mathematical sense. That is, there may be another route or routes that could be substituted and the same (maximum) flow would be achieved. If this is the case then there is actually an infinite number of solutions. If we have two solutions we can have an infinite number of linear combinations of the two. Operationally this is the same as saying that if we can add another route we can have an infinite number of variations by simply changing the levels of flow on the routes in our solution.

c. While there will be cases where the user is interested in alternative solutions in the mathematical sense, he will more often be interested in operational alternatives. That is, the military analyst may prefer an alternate solution that is not a mathematical optimum but has characteristics that have advantages not directly representable in the mathematical formulation of the problem.

d. A simple example may serve to illustrate this problem. The mathematical optimum solution to a problem may call for the heavy use of a link in the network that is considered to be vulnerable to enemy

action by the military analyst. Thus the military analyst may wish to reduce the flow over that link even though he knows that such a reduction will increase the cost of the overall movement.

e. Operational alternative solutions are readily obtainable using the ETNAM model. They do, however, involve re-runs of the model with the new restrictions incorporated in the system. In the example cited above, the capacity of the heavily used link would be reduced and the problem re-run. ETNAM has a re-start feature which permits a resumption of the algorithm at a particular point in the solution process. Thus, if an alternative solution is desired it is not necessary to "start from scratch."

CHAPTER 2. INPUT MODULE

1. Purpose and Scope. The primary purpose of the Input Module is to read data and to perform fundamental editing functions in structuring the data for the RACAT Module algorithms. This chapter describes the inputs required and outputs generated by the Input Module and discusses the methods used in the editing processes.

2. Inputs and Outputs.

a. Input Data Requirements. Two groups of input data are required to operate the ETNAM system. The first set must be in fixed sequence while the other may be entered in any order. Figure 2 is a schematic description of the input deck arrangement for the Input Module. This section describes the general formats for each type of input data. Enclosure 5 has some sample data listings.

(1) Sequenced Data. The Input Module has been designed to operate primarily upon input data which are not required to be in any fixed sequence for entry. The exceptions to this scheme which must be in sequence are:

Title Information
Run Parameters
Output Titles (in any order within this group)
Network Data (in any order within this group)
Node Constraint Data (if desired)

Each of these is discussed below.

(a) Title Information. The first data card is used as a title entry. All 80 columns are available and any alphanumeric entry is permitted. This entry becomes a header for each page of the reports in the Input, Output and POP Modules.

(b) Run Parameters. Entries for this card, which must follow the Title card, are as follows:

Columns	Information
1-8	Run objective (MAX FLOW, MIN COST, or MIN TIME)
15	Number of substitution groups allowed (≤3)
19-20	Number of transportation modes (≤20)
24-25	Number of commodities (≤20)
29-30	Number of resources (≤50)
34-35	Number of node constraints (≤50)
38-40	Type of link capacities 0 = some common unit of flow (e.g., short tons)

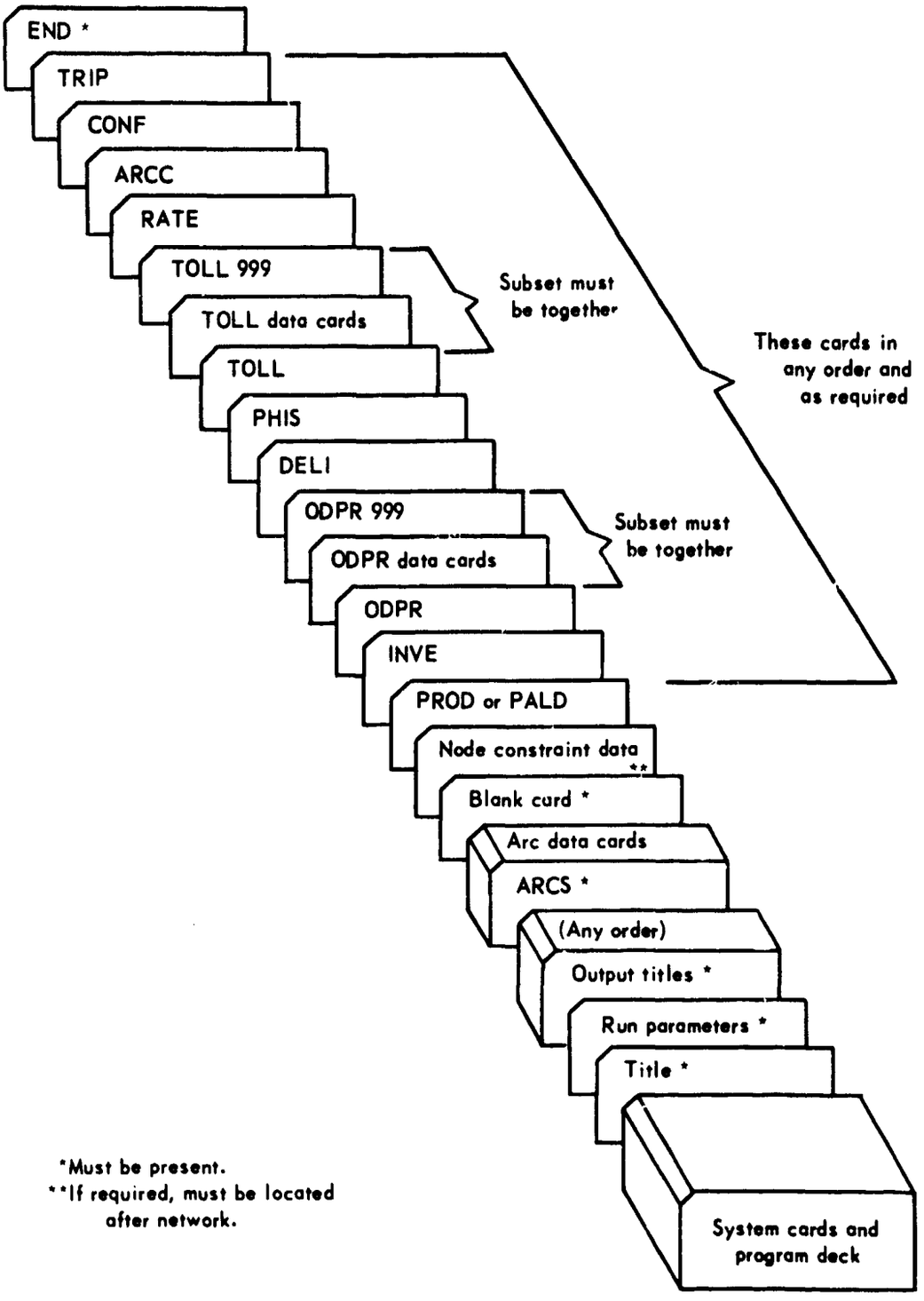


FIGURE 2. SCHEMATIC DESCRIPTION OF INPUT MODULE DECK

	1 = vehicles-per-time-period (V/TP)
43-45	Number of iterations before inversion (see RACAT MODULE, Chapter 4)
47-50	Maximum number of iterations permitted (see RACAT MODULE, Chapter 4)

(c) Output Titles. These cards must follow the run parameter card but may be entered in any order. The 4 characters in columns 1-4 identify the type of titles. Since the titles are assigned to modes, resources, and commodities by number, they should be entered in order on the card. The types of entry and dimensions are as follows:

Columns 1-4	Type of Titles	Dimension
NAMO	Names of modes	≤20
NACO	Names of commodities	≤20
NARE	Names of resources	≤50

and the format for all types is:

Columns	Information
1-4	Card type
10-13, 16-19, 22-25, 28-31, 34-37, 40-43, 46-49, 52-55, 58-61, 64-67	10 titles per card of 4 characters each (left adjusted)

(d) Network Data.

1. The first card of this set must be a title card with "ARCS" in columns 1-4. The network data cards follow and a blank card at the end signifies that all network data cards have been entered. It will be noted here that throughout the SD, the word arc is used interchangeably with the word link except where the specific word "ARC" or "ARCS" is used as input data. The format for data cards is as follows:

Columns	Information
1-6	FROM node, alphanumeric, left adjusted
7-12	TO node, alphanumeric, left adjusted
13-18	Mode of arc, integer
19-24	Condition of arc, real number
25-32	Distance of arc, real number

33-40	Capacity of arc, real number
50	Direction of arc (0 = undirected; 1 = directed)
51-70	Commodity availability (1 column for each commodity. 0 = will not handle; 1 = will handle)

2. The network data cards may be entered in any order since they are sorted in the Input Module. It will be noted again that the analyst has the option of stating the capacities of the links in terms of tons-per-time-period or in terms of vehicles-per-time-period. Whichever choice is made must be used consistently throughout the network.

(e) Node Constraint Data. If the capacity (either tons or V/TP) of certain nodes is to be constrained, the pertinent data must appear after the network listing. These data are stored in the following arrays:

NODCON (I)	=	Name of the constrained node I
NODMOD (I)	=	Number of the mode of constrained node I
RHS(NARC + NRES + I)	=	Throughput limit of constrained node I
Where NARC	=	The number of arcs
NRES	=	The number of resources

The throughput limits become right-hand-side entries for the constrained nodes. An error message is generated if more than 50 nodes are entered. The nodes which are origins or destinations are not allowed to be constrained. The format for these data cards is:

Columns	Information
1-6	Name of node (left adjusted)
7-10	Number of mode (right adjusted)
11-20	Throughput limit (node capacity constraint)

(2) Unsequenced Data. The remaining input data can be entered in any sequence since each card specifies in columns 1-4 the name of the type of data to be entered. Methods for entering each type of data are discussed below. Each card type is specified by a 4 character mnemonic. The general format for these data cards except origin/destination pairs and tolls is:

Columns	Information
1-4	Card type

6-10	Integer value	First set
11-15	Integer value	
16-20	Integer value	
21-30	Real value	
31-35	Real value	
36-40	Integer value	Second set (if desired)
41-45	Integer value	
46-50	Integer value	
51-60	Real value	
61-65	Real value	

The values in card columns 31-35 and 61-65 are used only for resource data and is discussed in that section.

(a) PROD or PALD - Resource Productivities.

1. Either PROD or PALD (but not both) may be used for any trio of commodity, mode, resource productivity. These data are stored in the arrays MASRES and RESMAS as follows:

MASRES (I, 1) = Number of commodity
MASRES (I, 2) = Number of mode
MASRES (I, 3) = Number of resource type
RESMAS (I) = Productivity for this type

where I is the index for the non-zero productivities.

2. There are two methods of inputing the productivities which are entered in their natural units (e.g., fraction of a bus per person per mile or fraction of a truck per ton per mile):

a. When the productivity is supplied by the user the PROD card will be used.

b. When the user supplies the payload of the resource and the total mileage traveled per day (including a return trip empty) for a typical unit of that resource, the model will calculate the productivity. This will require use of the PALD card. When the distance is greater than 1 the model will compute the inverse of the product of the payload and one-half the distance. If the distance is 1 (i.e., transfer units) the productivity is the inverse of the payload.

3. The productivities are entered sequentially into the matrix in any order. If no value is entered for a given trio of commodity, mode and resource, that productivity is assumed to be zero.

4. A choice of resources may be given to a commodity-mode pair. This is known as substitution of resources. ETNAM can accommodate up to three substitution groups. If a commodity-mode pair

is to have a choice of up to 3 sets of resources, each set is put into a different substitution group. But if another commodity-mode pair has no competitive resources, then each substitution group has the identical resource sets for that commodity-mode pair and must be repeated for each substitution group in the input data.

5. For purposes of inputing the data, the substitution group into which a resource productivity belongs needs to be identified. At most 50 resources may be used. These resources are numbered sequentially from 1 to at most 50 in substitution group 1. The same resources are numbered from 51-100 and 101-150 in substitution groups 2 and 3 respectively.

6. Error messages are produced if any subscript is blank or zero or if more than 700 entires are attempted. The format for these data cards is:

Columns	Information	
1-4	PROD	PALD
6-10	Number of commodity	Number of commodity
11-15	Number of mode	Number of mode
16-20	Number of resource	Number of resource
21-30	Productivity value	Payload
31-35	Blank	Total mileage (includes empty return trip)

(a second set if desired according to standard format).

(b) INVE - Resource Inventories. These data are stored in the array RHS. These data become right-hand-side entires for the resources. A subscript is required to identify the number of resource type. An error message is generated if the subscript is blank or zero or if more than 50 resource types are entered. The format for the INVE card type is:

Columns	Information
1-4	INVE
16-20	Number of resource
21-30	Total in inventory

(a second set if desired - columns 46-50 and 51-60).

(c) ODPR - Origin/Destination Pairs.

1. These data are stored in NORDES (I, 4), where I = number of O/D pair. These data are origin/destination pairings by

commodity number and a delivery requirement for that pair if appropriate. The delivery requirement is entered in the natural unit of the commodity. Normally a delivery requirement is imposed on an O/D pair. The alternative to that is to leave some or all O/D pairs for a given commodity "open" (i.e., do not specify a requirement) but specify a requirement for that commodity for the "open" O/D pairs in a DELI card (described below). The RACAT algorithm will assign appropriate flows to the "open" O/D pairs, the total of which is equal to the requirement specified in the appropriate DELI card.

2. If an O/D pair has a commodity delivery requirement of zero but is not to be an "open" requirement, that is, it is to have a requirement in a subsequent run, then the variable ID is set to 1. This effectively reserves a requirement row in the mathematical structure of the problem for subsequent use. If the pair is to be "open", ID is left blank and a DELI card must be present.

3. An error message is generated if more than 150 O/D pairs are entered or if any commodity number is >20.

4. The first card of this set must be a title card with "ODPR" in columns 1-4. The format of the data cards following it is:

Columns	Information
1-4	ODPR
7-10	Number of commodity, integer
14-19	Origin node, left adjusted
20-25	Destination node, left adjusted
26-35	Delivery requirement for this pair if any, a real number
36	ID = 0, if delivery requirement is present = 0, if delivery requirement is zero but the O/D pair is "open". = 1, if delivery requirement is zero but the O/D pair is not "open".

(a second set if desired in columns 37-40, 44-49, 50-55, 56-65, 66).

The last card must have "ODPR" in columns 1-4 and "999" in columns 8-10. All integers in the above format are right adjusted in their respective fields.

(d) DELI - Delivery Requirements. These data are stored in the array DELREQ (I), where I = number of commodity. Any O/D pairs left "open" need a total delivery requirement by commodity. Closed O/D pairs do not have corresponding DELI cards. An error message is

produced if a commodity number is blank, zero, or >20. The format for the DELI card type is:

Columns	Information
1-4	DELI
16-20	Number of commodity
21-30	Total delivery requirement for this commodity.

(a second set if desired - columns 46-50 and 51-60).

(e) PHIS - Prices on Resources. These data are stored in the array PHI (I), where I = number of resource type.

1. Prices on resources are required to operate the model in a min-cost objective. Resource prices must not be used for the max-flow or min-time operating objectives. An error message is generated if the resource type number is blank, zero, or greater than 50.

2. Prices are entered in their actual dollar cost form. The Input Module normalizes these prices by searching for the minimum non-zero price and then converting all the actual prices to a relative scale. The format for the PHIS card type is:

Columns	Information
1-4	PHIS
16-20	Number of resource
21-30	Price

(a second set if desired - columns 46-50 and 51-60).

(f) TOLL - Tolls on Certain Arcs.

1. These data are stored in the array TOLLS. Tolls may be placed on selected arcs by specifying three items: mode; FROM node; and TO node. The program will locate the desired arc row and impose the specified toll on the particular arc. An error message is produced if the specified arc cannot be located in the arc matrix.

2. The format for the TOLL card is similar to the ODPR cards. The first card of this set must be a title card with "TOLL" in column 1-4. The format of the data cards following it is:

Columns	Information
1-4	TOLL
7-10	Number of arc mode, integer-right adjusted

Columns	Information
14-19	Name of FROM node, left adjusted
20-25	Name of TO node, left adjusted
26-35	TOLL to be imposed, a real number

(a second set if desired in columns 37-40, 44-49, 50-55, 56-65).

The last card must have "TOLL" in columns 1-4 and "999" in columns 8-10.

(g) RATE - Rates of Speed. These data are stored in the array RATES (I), where I = number of mode. Rates of speed by mode may be entered in order to calculate delay times on the arcs. If the model is to be operated in a minimum time objective, these data are required. An error message is generated if the number of mode is blank, zero, or greater than 20. The format for the RATE card type is:

Columns	Information
1-4	RATE
16-20	Number of mode, integer-right adjusted
21-30	Speed, a real number

(a second set if desired - columns 46-50 and 51-60).

(h) ARCC - Arc Data Update. This feature is discussed under the section "Methods" on performing network data updates. The format for the ARCC card type is:

Columns	Information
1-4	ARCC
11-15	Data field
16-20	Constant to be added
21-30	Multiplier
36-40	Limit value
41-45	Data field
46-50	Limit value

(i) CONF - Conversion Factors.

1. These data are stored in the array CONF (I), where I = commodity number. They must not be used when node and link capacities are expressed in vehicles-per-time-period.

2. When the link and node capacities are expressed in some equivalent unit like short tons-per-time-period, then conversion factors are specified for any commodity not expressed in the same units as the capacities. Normally, the capacities are specified in short

tons-per-day; thus any commodity such as passengers or POL will need to be converted to short tons to be moved over the network. Both the delivery requirements and the productivities are converted by the Input Module.

3. The format for the CONF card type is:

Columns	Information
1-4	CONF
16-20	Number of commodity
21-30	Conversion factor

(a second set if desired in columns 46-50, 51-60).

(j) TRIP - Payload factor.

1. If the link capacities are stated in vehicles-per-time-period these data cards must be present. The data are stored in the array TRIP (I, J, K), where:

I = number of substitution group
J = number of commodity
K = number of mode

2. The factors are the normal payloads of a vehicle (resource) by commodity by mode for each substitution group. If substitution groups are used, the factors for substitution groups 2 and 3 need not be entered unless they differ from substitution group 1. If a certain commodity is not allowed on a mode, i.e., passengers on a pipeline, no factor is entered, but for all allowable commodity/mode combinations there must be a factor.

3. An error message is generated if any commodity or mode number is blank, zero, or >20. The format for the TRIP card type is:

Columns	Information
1-4	TRIP
6-10	Number of substitution group
11-15	Number of commodity
16-20	Number of mode
21-30	Trip payload factor

(a second set if desired according to standard format).

(k) END. Signifies End of Input Data. This must be the final card in the input data file. (Columns 1-3 = END). An error message

is produced if a data card does not contain one of the above names for the type of input in columns 1-4. Figure 2 is a schematic description of the input deck arrangement for the input module.

b. Output Reports. The input data reports of the Input Module are generated by the WRITES routine. The formats used in these reports are described as follows:*

(1) Title and Run Parameters Report.

(2) Rates of Speed by Mode Report. Identifies the modes and their rates of speed. Up to 20 modes are permitted and will be reported.

(3) Commodity Identification Report. It identifies each commodity and gives its conversion factor if any. Up to 20 commodities are permitted.

(4) Network Data Listing. This report is produced after the arcs are sorted on the FROM node. The "SYM and Commodities" is a packed-word output in octal format on the CDC 6400 and in hexa-decimal format for the IBM System/360.

(5) Table of Uniq Nodes.

(6) Node Constraint Report. Up to 50 node constraints are permitted. The report will appear only if node constraints are used.

(7) Resource Inventories. Up to 50 resource types are permitted and will be reported.

(8) Resource Productivities Report. These reports are generated for each substitution group by commodity by mode.

(9) Trip Payload Report. This report is generated for each substitution group only if vehicles-per-time-period link constraints are used.

(10) Origin-Destination Data Report for a Minimization Run. A maximization run would not show delivery requirements.

c. Output File. The output file of the Input Module becomes the input file for the RACAT Module. It is a binary file and is in the following sequence:

(1) Title and Run Parameters.

---

*See Enclosure 1 of SD, Volume I, User's Manual, for examples of computer-produced reports based on a test problem.

(2) Network Data.

(3) Node Constraints (if any).

(4) Resource Inventories.

(5) Resource Normalized Prices and Smallest Non-Zero Price (Min-cost operating objective only).

(6) Origin/Destination Pair Indexes for List Processing.

(7) Origin/Destination Pair Data.

(8) Delivery Requirements (minimization objective only).

(9) Substitution Indexes for List Processing.

(10) Bit Map for Resource Productivities Substitution Changes (if substitution is used).

(11) Resource Productivities.

(12) TRIP Payload Factors (if link capacities are in V/TP).

(13) Conversion Factors for Commodities.

(14) Output Titles (for Output Module only).

(15) Table of Unique Nodes (for Output Module only).

3. Methods. The logic utilized by the Input Module consists basically of eight phases of operations (or subroutines):

Read input data (INEDIT).
Sort and edit data (SORTAR).
Set-up origin/destination pairs (ODROWS).
Write input reports and file for RACAT Module (WRITES).
Convert requirements and productivities (CONVER).
Set up table of unique node names (NODIDX).
Perform network update (MODIFY).
Perform and report on error checks (ERROR).

Each phase is discussed below and is shown in the Input Module Functional Flow Chart, Figure 3.

a. Read Input Data (INEDIT). The functions of the INEDIT routine are to enter the input data specified in the previous section into the appropriate arrays and to monitor the editing subroutines.

(1) Basic network data are entered and stored in the following ways:

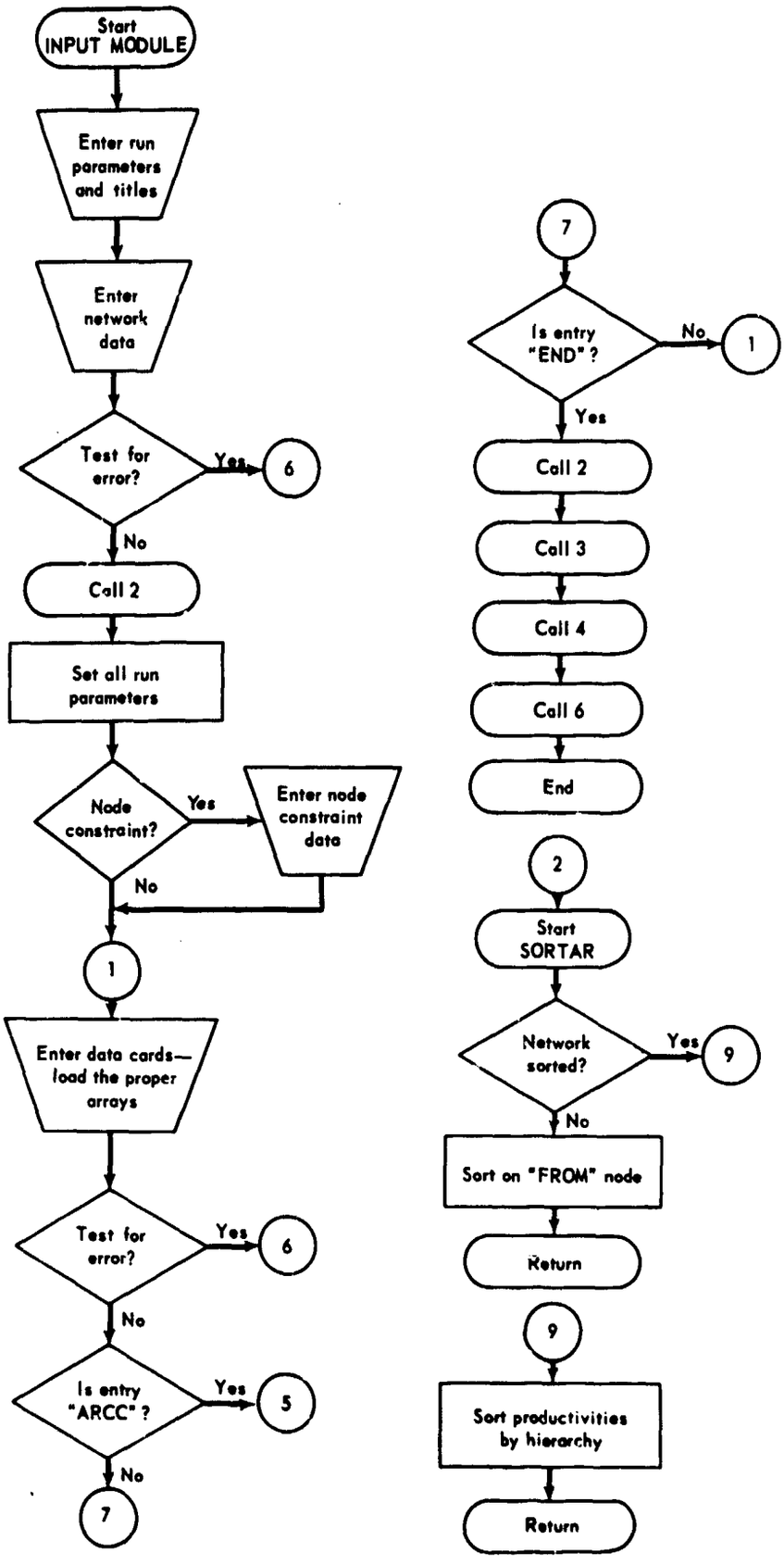


FIGURE 3. INPUT MODULE FUNCTIONAL FLOW CHART

(Sheet 1 of 3)

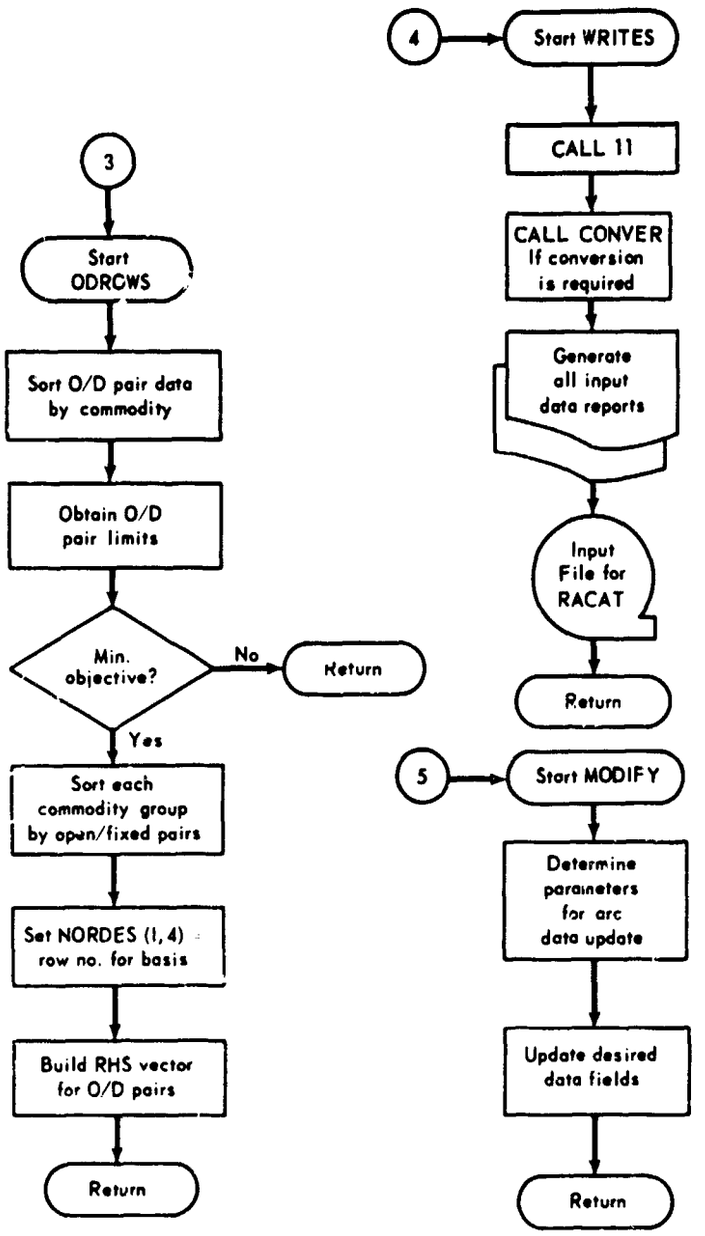


FIGURE 3. (continued)
(Sheet 2 of 3)

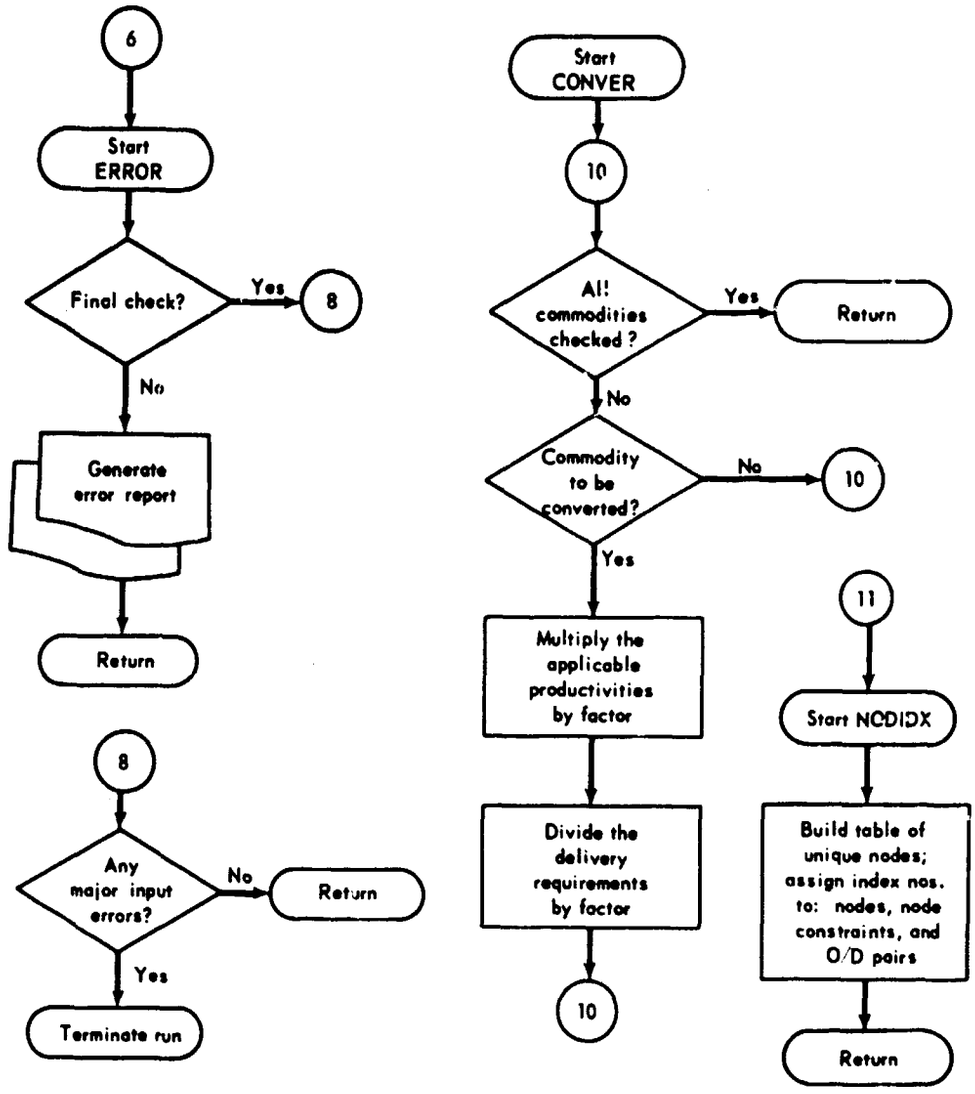


FIGURE 3. (continued)
(Sheet 3 of 3)

I	= Arc No.
MATARC (I, 1)	= FROM node
MATARC (I, 2)	= TO node
MATARC (I, 3)	= Mode of arc
COND (I)	= Arc condition
COST (I)	= Arc cost (distance)
RHS (I)	= Arc capacity
I1	= Direction (0 = undirected; 1 = directed)
NCD (20)	= Commodity availability (0 = will not handle commodity k; 1 = will handle commodity k).
NCOMMD (I)	= The array in which the direction (bit 21) and the commodity availability (bits 1-20, right to left in the CDC 6400 and right to left in the IBM/360) bits are stored for arc I.

( ) A packing technique is used for the arc direction and commodity availability matrix in order to minimize core utilization. The bits pertaining to arc I are packed in NCOMMD (I). The first 20 bits of a word are used for the 20 commodities; i.e., commodity 1 = bit 1, etc. If the particular arc can accommodate commodity k then bit number k of the word is turned on (set to 1); if it cannot, it is left off (zero). An additional bit is used to specify arc direction. The 21st bit of a word is for symmetry. If the arc is directed, the bit is turned on (set to 1); if the arc is undirected, it is left off (zero). See Chapter 3 for more details.

(3) Delay times on the arcs are computed by the equation

$$\text{TRTIME (I)} = \frac{\text{COND (I) x COST (I)}}{\text{RATES (J)}}$$

where,

I = Arc row number
J = Mode of the arc

(4) Run parameters are set in the INEDIT subroutine according to the following dimensions:

NARC	= Number of arc rows (≤700)
NRES	= Number of resource constraints (≤50)
NODR	= Number of movement requirement rows (≤150)
NROW	= Total number of rows in the basis (≤900)
NCONST	= Number of node constraints (≤50)

Note that NROW = NARC + NRES + NODR + NCONST could equal 950 if all were at their maximum. This is not allowed. NROW is restricted to be ≤900.

b. Sort and Edit Data (SORTAR). Some elementary editing operations are performed on the input data upon completion of reading the network data and all other input data.

(1) Network Data Sorting. An internal sort routine is used to sort the entire network data on the FROM node. The sorting process is used to increase the efficiency of the RACAT Module. The sort routine used is a "Shell Sort" (developed by C. Shell) modified to accommodate the special data. This routine is extremely fast and effectively has as its maximum number of comparisons:

$$[N \log_2 N]$$

where N = Number of elements of the array to be sorted.

Run indices are computed based on the input data as follows:

MP = NARC + NRES
NODROW = MP + NCONST
NROW = NODROW + NODR

(2) Resource Productivities Sorting. The modified Shell Sort routine is used again to sort the resource productivities. The sort hierarchy is as follows:

Substitution group

Commodity number

Mode number

Type of resource.

Thus the MASRES matrix is sorted as is the RESMAS vector containing the resource productivities to correspond with the proper elements. Indices are established to define the first and last entry of each substitution group. The resource type numbers in substitution groups two and three are reduced to the range 1-50 by subtracting 50 and 100 respectively. This yields the actual index of the resource.

(3) Resource Cost Comparisons. The cost of the resources (productivity multiplied by the price) in each substitution group is compared. The least cost resource is placed in group one, second in group two, most costly in group three. Other data which are given by substitution group are rearranged accordingly. This is done to aid the RACAT Module in obtaining a lower priced feasible solution at the end of Phase I than it would otherwise during a min-cost run with substitution of resources. The resource prices have no influence on the Phase I objective function. Consequently, the first available resource set is used. With the resources rearranged as described here, the cheapest resource set will be used first.

c. Set Up Origin/Destination Pairs (ODROWS).

(1) The function of subroutine ODROWS is to manipulate the origin/destination (O/D) pair data and to establish the appropriate rows in the basis for a minimization run. First the O/D pair data are sorted by commodity. The storage assignments are as follows:

NORDES (I, 1) = Commodity number
NORDES (I, 2) = Origin node
NORDES (I, 3) = Destination node
NORDES (I, 4) = Row number for basis (for minimization run)

(2) The "fixed" delivery requirement for each pair, if specified, is entered into DELREQ (20 + I), where I = number of O/D pair. The first 20 words of DELREQ are reserved for any "open" delivery requirements for the ≤20 commodities. Thus an O/D pair may be "open", where a delivery requirement is not specified, or "fixed" where a delivery requirement is specified in the input data.

(3) The limits of the O/D pairs are then obtained and stored in (NCD (I), I = 1, 20) for the ≤20 commodities. These indices are used to process the list in the RACAT Module.

(4) For a minimization run, the O/D pair data are sorted within each commodity by "open" and "fixed." "Open" O/D pairs are assigned a row number in the basis corresponding to the particular commodity. The right-hand-side delivery requirement for this row is the amount to be delivered for the "open" O/D pairs of that commodity. The "fixed" O/D pairs are assigned individual row numbers in the basis since they have specific right-hand-side delivery requirements. The right-hand-side vector for the O/D pair rows is then built on the basis of these row assignments.

d. Write Input Reports and File for RACAT Module (WRITES). This routine is used to generate the reports and file discussed in the previous section under "Output Reports" and "Output Files". The methods used are straight-forward report routines.

e. Convert Requirements and Productivities (CONVER).

(1) This routine performs the conversion of delivery requirements and productivities for all commodities not normally specified in the same units as the link capacities. The conversion is not made if the capacities are expressed in vehicles-per-time-period (V/TP), since in that formulation the commodities are accepted in their natural units. Commodities such as passengers and POL are better specified in their natural units as people and barrels respectively. When not in the V/TP formulation, it is still convenient to input the commodities and

their corresponding productivities and movement requirements in terms of their natural units. Thus a conversion factor is used by the program to equate all the commodities to a single equivalent unit - usually short tons.

(2) The delivery requirements for a commodity requiring conversion are divided by the specified factor. For example, 70 passengers to be moved with a conversion factor of 7 would be set equal to 10 equivalent short tons by CONVER.

(3) Resource productivities applying to any commodity requiring conversion are multiplied by the specified factor. For example, a troop carrier productivity of .001 (40 men, 500 miles per day including a return trip empty) would be converted to .007 vehicles per ton mile.

f. Perform Network Update (MODIFY).

(1) As mentioned earlier, the ARCC data card provides the capability of updating certain network data. There are 8 "fields" of arc data that are defined as follows:

Field	Description
1	FROM node
2	TO node
3	Mode of arc
4	Condition of arc
5	Cost (distance) of arc
6	Toll of arc
7	Capacity of arc
8	Direction of arc

The desired update of most of the above fields is accomplished by the use of a linear combination. The FORTRAN expression is

$$[F_i^*] = Q \cdot [F_j] + X$$

where

$[F_i^*]$ = The data field to be modified

$[F_j]$ = Any data field (i = j is acceptable)

Q = Multiplier

X = Value to be added

Thus the new value of field i is set equal to the old value of field j times a multiplier plus some value.

(2) Provision is also made for specifying limits between which the updating is to be accomplished. This is performed by the inequality:

$$Y \leq [F_i] \leq Z$$

where

$[F_i]$ = The data field to which the limits apply
Y = The lower limit for modification
Z = The upper limit for modification

Thus one may update field F_i for the entire network or within specified limits as indicated by the inequality. The $[F_i]$ in the inequality may refer to another field if desired.

(3) For example, assume it is desired to impose a condition of 1.0 to all arcs of the network which are mode 6. We would use both equations as

$$F4 = 0. \cdot F4 + 1.0$$

and

$$6 \leq F3 \leq 6$$

which would set data field 4 (condition of arc) equal to zero times field 4 plus 1.0 for all arcs which meet the criterion that data field 3 (mode of arc) have a value ≥6 and ≤6.

(4) The above expressions are applicable to any combination of the third through eighth network data fields. Fields 1 and 2 cannot be used because they are alphanumeric fields (FROM and TO nodes). If the updating is performed on the integer field, field 3 (mode), the new value is rounded.

g. Set Up Table of Unique Node Names (NODIDX). The NODIDX routine finds the unique nodes (there must be at least four in the network for the binary search technique to work properly) and places them in the array LABTAB (J). The index number of the array is then placed in MATARC (I, 1) and MATARC (I, 2), the FROM and TO nodes, and written on the file, TAPE1, for use in the RACAT Module. An error message is generated if the array is >400. The nodes that are constrained and the O/D pair nodes are assigned their respective index numbers. While both the Input and Output Reports print alphanumeric node names, RACAT uses integer numbers.

h. Perform and Report on Error Checks (ERROR).

(1) An input data error routine is used to report the major

input errors and, if necessary, terminate the run. The data processing continues until all data are completed in order to report all the discovered errors in one pass. The specific error messages relating to each type of input data are mentioned in the section in this chapter on "Read Input Data".

(2) In addition to the errors by card type, the routine checks for possible run parameter limits, proper amounts of output titles, and excessive row entries and will generate error messages if appropriate. The error messages are reported before the normal input data reports are produced. Upon completion of the input data reports, the error routine decides whether any discovered error is a major one and, if so, terminates the run.

(3) Several self-explanatory error messages are reported in other routines.

CHAPTER 3. RACAT MODULE

1. Purpose and Scope. The purpose of the RACAT Module is to solve the multi-commodity network flow problem with mobility resource allocation as formulated by the Input Module in the data file. The technique permits the maximization of flow or the minimization of cost (or time) subject to the constraints of the capacity of the network and the availability of resources. Due to the mathematical sophistication of the algorithms, this chapter will be confined to a discussion of the general methods used to solve the problems. Specific information with respect to the algorithms will be found in Enclosures 1, 2 and 3.

2. Inputs and Outputs. Inasmuch as the RACAT Module is a set of algorithms for solving specific problems, the inputs and outputs are well-defined and structured so as to facilitate efficiency toward arriving at solutions. This section describes the input data requirements and output files generated by the RACAT Module.

a. Input Data Requirements. There are two types of input data required to operate the RACAT Module: first, the data file (TAPE1) written by the Input Module; and secondly, one card specifying certain parameters and the mode of operation to be used -- NORMAL or RESTART. However, the restart procedure requires more data and will be discussed later.

(1) Data File. The contents and order of the data file are specified in Chapter 2, INPUT MODULE. The file name is TAPE1 and is written in binary logical records. As discussed in Chapter 2, the data are structured specifically for RACAT.

(2) RACAT Data Card. One punched card is required to operate the RACAT Module and is used to specify the operating mode and optional control parameters. Two modes of operation are possible - NORMAL and RESTART. These are discussed in detail under the section of this chapter entitled "Methods." The format of the RACAT data card and the definitions of the parameters are as follows:

Columns	Information
1-7	NORMAL or RESTART
11-15	NITER
16-20	INVC
21-25	IXPRNT
26-30	NREPRT
31-35	ITSAVE
36-40	IREST
41-45	IRTARC
46-50	IRTRES
51-55	ISIMP

NITER Maximum number of iterations until job is terminated.

INVC	Frequency of inversion.
IXPRNT	Frequency of debugging print.
NREPRT	Iteration number when solution report is to be obtained (before optimum) and job terminated.
ITSAVE	Frequency of saving basis and bit map on TAPE9.
IREST	Equal to 0, 1, 2 or 3 0 = Normal restart from file and mark slacks. 1 = Restart from file and do not mark slacks. 2 = Restart from cards and mark slacks. 3 = Restart from cards and do not mark slacks.
IRTARC	Equal 0 or 1 0 = Do not print arcs on iteration log. 1 = Print arcs on iteration log.
IRTRES	Frequency of printing resources on the iteration log.
ISIMP	Frequency of printing simplex multipliers.

The parameters on the RACAT data card are optional; if not desired, the appropriate data field is left blank or set to 0. NITER and INVC may have already been set in the Input Module. Those values may be retained by leaving the appropriate data fields blank or setting them to 0. They may be reset by specifying their new values on the RACAT data card.

b. Outputs. Four output files are generated by the RACAT Module.

(1) The data file (TAPE1) is used to output certain information obtained by the optimal solution. The data listed below are added to the data file (TAPE1) when an optimal solution is reached. These are explained in detail under "Methods" in this chapter.

(a) The vector identification numbers present in the basis at optimization.

(b) The solution vector.

(c) The simplex multipliers (shadow prices).

(d) The value of the objective function.

(e) The number of iterations required to reach the solution.

(2) In addition, a vector file (TAPE2) is created during the algorithm which specifies the vectors as they are generated to enter the basis. The creation and maintenance of the vector file is discussed under "Methods" in this chapter. TAPE2, a binary file, is structured as follows:

(a) Index number of this vector

(b) Origin/destination number of this vector.

(c) Row number of delivery requirement intersect.

(d) Chain delay time.

(e) Number of links in the chain.

(f) Link row numbers in the chain.

(g) The vehicle payload factors (if link capacities are expressed in V/TP).

(h) Resource requirements per unit flow on this chain.

(i) Constrained nodes in the chain, if any.

(3) A save-basis file (TAPE9) is created for restart purposes. This file is explained under the Restart section of "Methods." TAPE9, a binary file, is structured as follows:

(a) The number of structural vectors (or routes) in the basis.

(b) The identifying number of each structural vector in the basis.

(c) The bit map identifying the slack vectors in the basis.

(4) A POP file (TAPE14) is created for use by the Post Optimal Package Module. This file is a binary file. It is discussed further in Chapter 5. During the RACAT computer run, TAPE14 is used as the eta file, where the product form of the inverse is stored if it cannot be contained in core. The structure of the file TAPE14 at a normal ending of a RACAT run is as follows:

(a) The product form of the inverse of the basis (count parameters, bit map and packed eta vectors with the zeros removed).

(b) Type of record indicator, run title, necessary parameters (NARC, NRES, NCONST, NROW, ITER, ISORTI, NCOM) and conversion factors.

(c) Right-hand-side values.

(d) Vector identification numbers present in the optimum solution.

(e) The solution vector.

(f) O/D data.

3. Methods. The RACAT Module set of algorithms for solving the multicommodity network flow problem with resource allocation has been designed to operate in either a maximization of flow or a minimization of cost (or time) objectives. To accomplish the optimization, a series of 18 functional subroutines is used. Each of these subroutines is discussed in this section. These discussions are general in nature and describe the algorithms in a functional framework.* Three bit manipulating subroutines are also used and are briefly described.

a. Subroutines. These functional subroutines are discussed in this section by describing the algorithms for solving the two operating objectives of the model - maximization of flow and minimization of costs -- and defined in terms of their primary function as follows:

(1) RACAT. Reads and loads the input data file.

(2) SETUP. Sets up the algorithms for the appropriate operating objective.

(3) PHASE1. Obtains a feasible solution to a minimization problem (Phase I).

(4) SMPLEX. Generalized product form of the inverse linear programming algorithm.

(5) ALPHAS. Computes simplex multipliers.

(6) MAXIO. Controls shortest chain algorithm for a maximization problem.

(7) MINIO. Controls shortest chain algorithm for Phase II of a minimization problem.

(8) SUBST. Computes resource vector each iteration for shortest chain costs.

(9) SCHAIN. Shortest chain algorithm for column generation subproblem.

(10) SORTAJ. Internal sort for inversion routine.

(11) REINV. Operates inversion routine.

(12) RESTART. Operates restart operation to introduce vectors into the basis.

---

*See Enclosures 1, 2 and 3 for the mathematical derivation of the RACAT Algorithm.

(13) WRITE. Generates output at optimum.

(14) ERROR. Terminates RACAT due to some major error during algorithm.

(15) NODEC. Apportions the simplex multipliers of the constrained nodes to the appropriate link.

(16) SBASIS. Saves the basis.

(17) SORTIE. Supplies the vehicle payload factors for constrained nodes and the shortest chain.

(18) ETAFLE. Manages the basis inverse in product form and packed format. The basis inverse is modified from iteration to iteration. The modification is in the form of a multiplication by an elementary column matrix. The elementary column matrix has one column which differs from the identity matrix of the same dimensions. This column is known as the eta vector corresponding to that elementary column matrix. Only the index giving the position of the eta vector in the elementary column matrix and the eta vector itself need be stored every iteration. Usually the eta vector is very sparse, that is, it has many zero entries. ETAFLE stores only the non-zero numbers along with a bit map indicating their positions in the eta vector. It also retrieves these numbers when needed and makes the necessary computations of only the non-zero numbers as appropriate each iteration to conserve computer time. ETAFLE is discussed in detail at this point because its function is similar for all operating objectives. It is called by the subroutine ALPHA in computing the simplex multipliers, by SMPLEX in removing and entering a variable into the basis and generating the corresponding eta vector. REINV and RESTART also use it to generate a new eta file.

(19) IONBIT, IOFFBIT, KBIT. These are bit manipulating subroutines. The arguments for all three are the same, (NWORD, NUMBIT). IONBIT sets the bit in NWORD given by NUMBIT to 1. IOFFBIT sets it to 0. KBIT tests that bit whether it is on or off.

b. Maximization Objective. The simplest operating objective is that of flow maximization. Figure 4 is a simplified flow chart of the RACAT Max-Flow Algorithm. The problem as formulated in words is to determine the maximum flow over all possible chains from origins to their respective destinations subject to the capacity constraints of the arcs and the availability of resources. The RACAT Module is designed to solve the maximization problem by an iterative sequence of routines to accomplish the algorithm as specified in Figure 4. This sequence is discussed below. Figure 5 is a general flow chart of the RACAT Module max-flow process showing the functional subroutines sequence for a normal run. The functions of each subroutine are as follows:

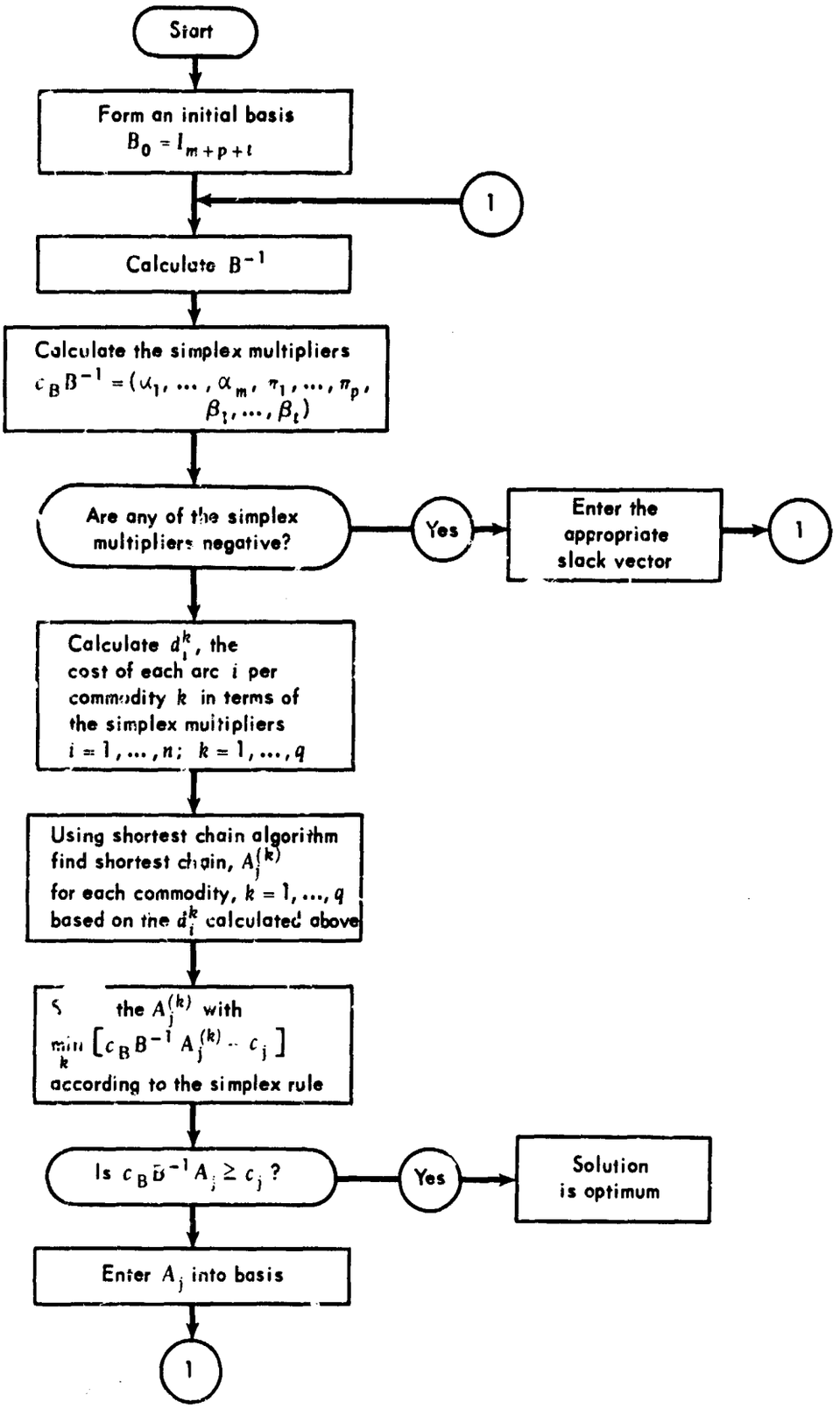


FIGURE 4. SIMPLIFIED FLOW CHART OF RACAT MAX-FLOW ALGORITHM

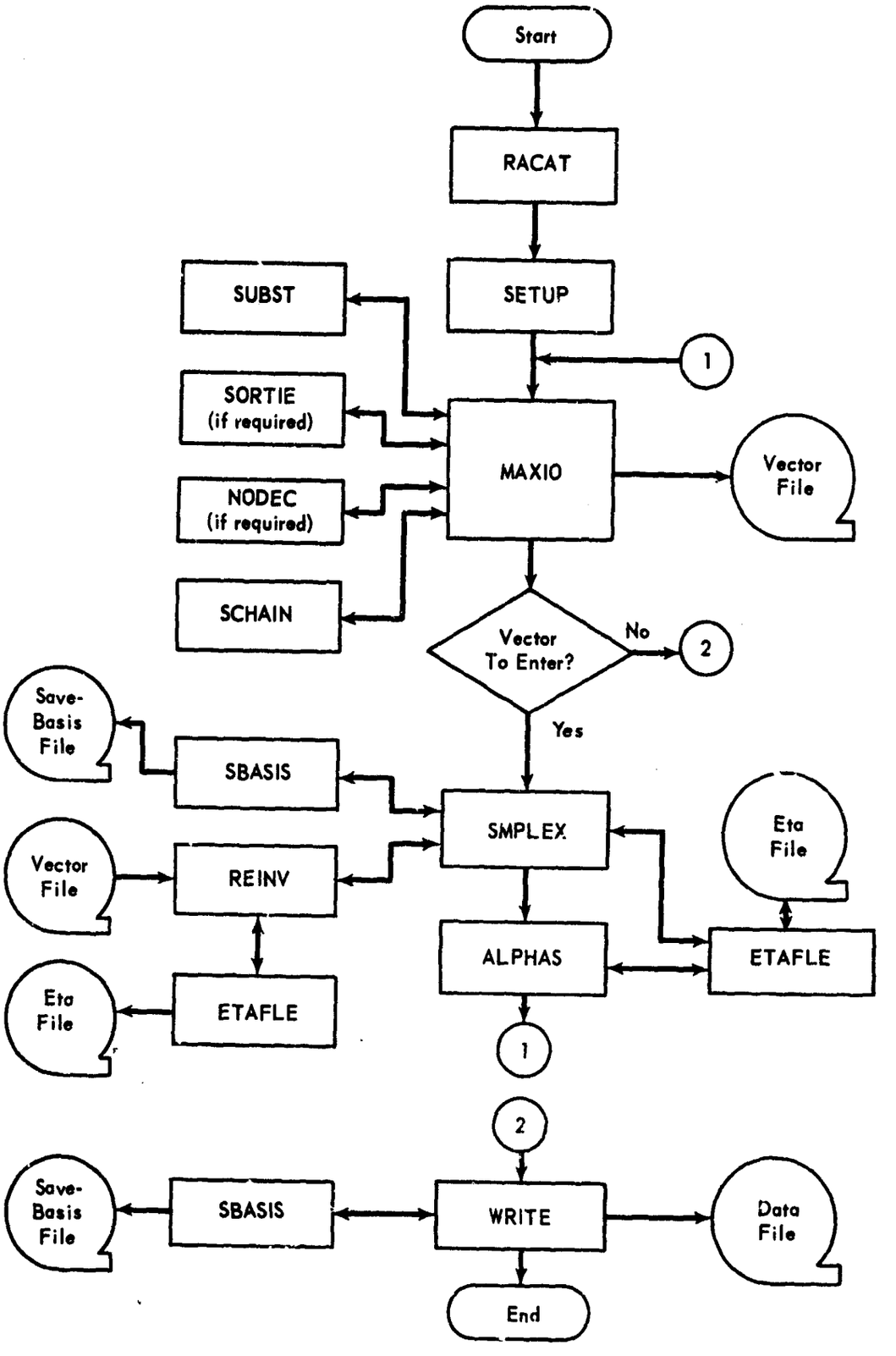


FIGURE 5. GENERAL FLOW CHART OF RACAT MODULE MAX-FLOW PROCESS

(1) RACAT. The RACAT subroutine is used to load the input data file into the proper storage locations. It reads TAPE1, the data file created by the Input Module discussed in Chapter 2. No editing is performed since the Input Module structures the input file specifically for RACAT. It sets up various tables of the network data to be used by the shortest chain algorithm.

(2) SETUP. This subroutine performs initialization of the appropriate variables, vectors, and matrices and then sets up the operating objective. The algorithm starts with an identity matrix as a beginning basis and a solution vector equal to the right-hand-side values. The number of rows for a maximization is the following sum.

NROW = NARC + NRES + NCONST

where,

NARC = Number of arc constraints

NRES = Number of resource constraints

NCONST = Number of node constraints.

(3) MAXIO. This subroutine operates as the controller for a maximization run.

(a) Its functions each iteration (see Figure 6) are delineated as:

1. Initialize variables.
2. Access an origin/destination pair.
3. Call SUBST routine for appropriate resource vector.
4. Assign pseudo-costs to arcs of the network.
5. Call SCHAIN routine to generate a vector.
6. Call SORTIE routine if V/TP constraints formulation is used.
7. Call NODEC routine if node constraints are present.
8. Evaluate vector to enter basis against simplex criterion.
9. Repeat steps 2-6 for all O/D pairs to obtain the "best" vector to enter the basis.
10. Write this vector on the vector file.

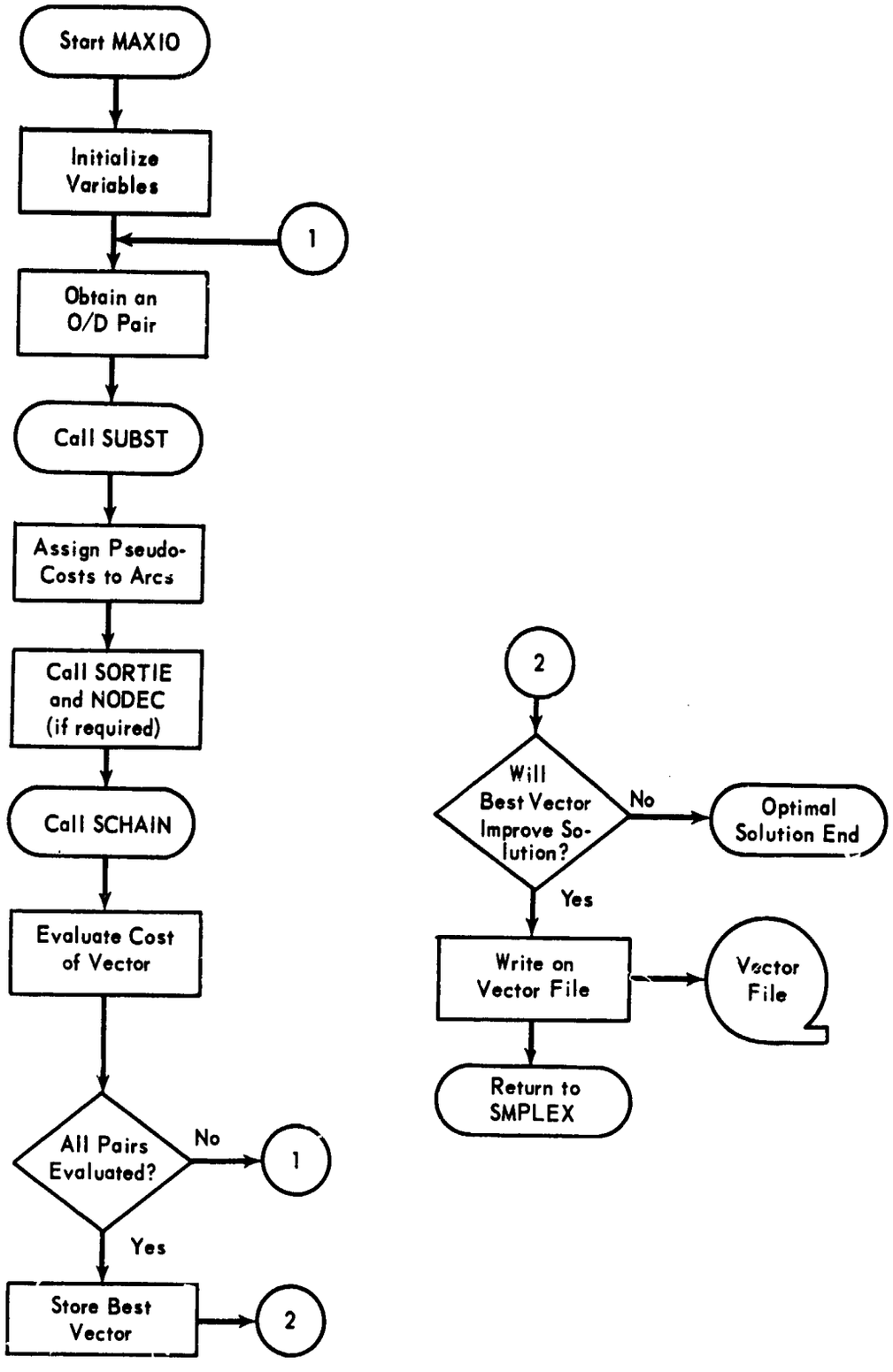


FIGURE 6. GENERAL FLOW CHART OF MAXIO ROUTINE

11. Call SMPLEX routine to update basis.

12. If no vector meets the criterion, terminate algorithm.

(b) Each vector file record is written as follows:

1. NAJ = Number of vector. A sequential numbering scheme beginning with NROW + 1.

2. NP = Number of the origin/destination pair associated with this vector.

3. NR = No useful information for the "Maximization Objective."

4. CTME = The delay time for this chain.

5. NAR = Number of arcs contained in the chain.

6. (AJ(I), I = 1, NAR) = Arc row numbers in the chain.

7. (AJ2(I), I = 1, NAR) = Vehicle payload factor for the arc whose row number is in the chain.

8. (COLUMN(I), I = NARC + 1, NODROW) = Resource coefficients and constrained nodes for this chain.

(4) SUBST. This subroutine performs the necessary calculations each iteration to compute the appropriate resource vector for each transportation mode for a commodity number. If ISUB = 1 (i.e., no substitution alternatives permitted) the computation involves merely processing the master list of resource productivities (MASRES, RESMAS) and building the appropriate vector. If ISUB = 2 or ISUB = 3, each feasible set of resource productivities is evaluated against the current simplex multipliers to obtain the best alternatives from which to generate pseudocosts for the arcs. Figure 7 shows a general flow chart of the SUBST routine used in the maximization operating objective.

(5) SCHAIN. This subroutine consists of a shortest chain algorithm described by Dreyfus[1] as the computationally most efficient procedure and credited to E. W. Dijkstra. The problem is that of finding a minimal cost chain from one node to another in a network in which each arc (i, j) has an associated cost (or length) $d(i, j) \geq 0$.

[1]See Dreyfus, S. E., An Appraisal of Some Shortest-path Algorithms, Rand Corporation, Memo. RM-5433-1-PR, September 1968.

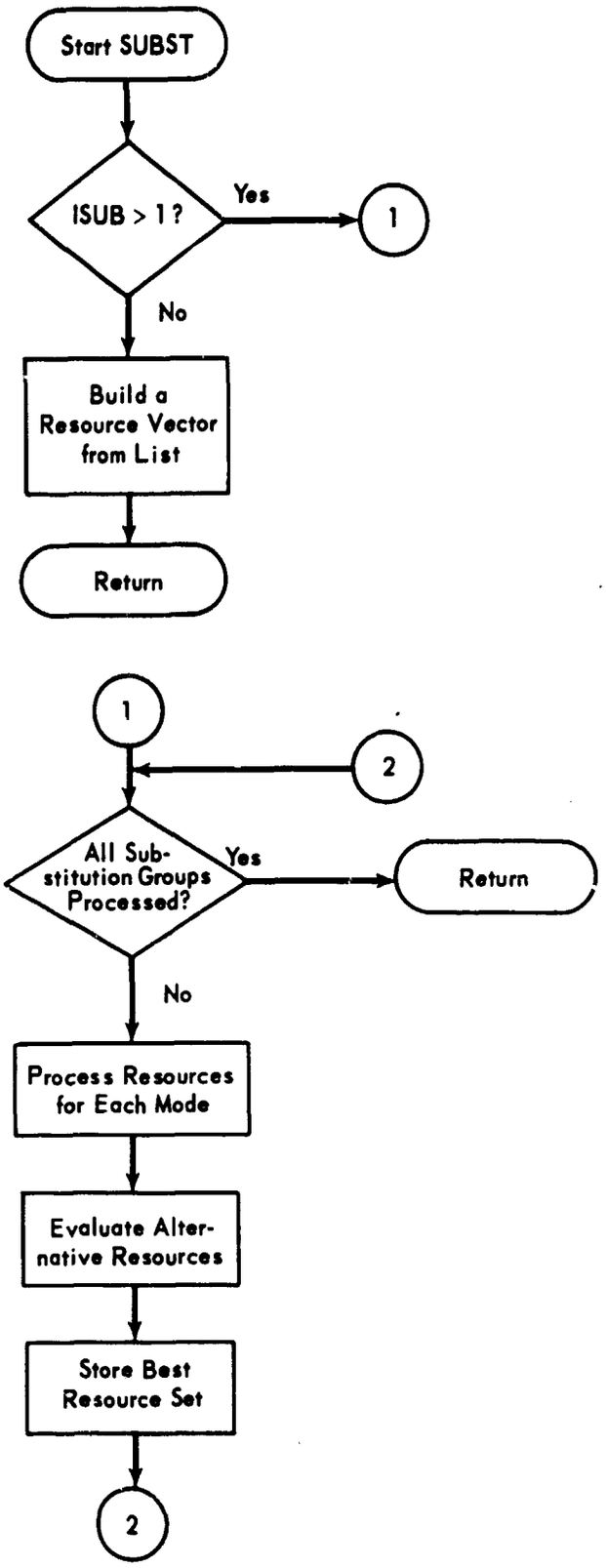


FIGURE 7. GENERAL FLOW CHART OF SUBST ROUTINE--MAXIMIZATION OBJECTIVE

Thus the restriction that all costs $d(i, j) \geq 0$ precludes the possibility of negative cycles and enables the algorithm to find the minimal cost chain.

(a) Figure 8 illustrates the shortest chain algorithm used in the SCHAIN routine. The algorithm has been modified to choose the chain with the minimum distance in case of ties. This method has in most cases proved to be more efficient with respect to speed in obtaining an optimal solution to the larger problem. An alternate method is to choose the chain with maximum capacity in case of a tie. This method is especially suited for a max-flow problem. Thus the problem structure determines which of the two methods is best. The origin and destination nodes are specified and the pseudocosts applied to the arcs are computed in the MAXIO routine.

(b) The SCHAIN routine checks each arc before it is labeled to insure the arc can accommodate the commodity number with which the origin/destination pair is associated. This procedure is accomplished by inspection of the appropriate bit as "packed" into a word which is discussed in Chapter 2, INPUT MODULE. If the appropriate bit is "on," the arc can accommodate the particular commodity and is a candidate for the shortest chain; if the appropriate bit is "off," the arc cannot accommodate the commodity and thus is not labeled. Checking for directed arcs is handled similarly.

(c) An error message is produced and the algorithm is terminated if no chain can be found between any origin and destination. This indicates a deficiency in the network data and results in termination of the RACAT Module.

(6) SMPLEX. The SMPLEX subroutine is a linear programming code based on the product form of the inverse. The algorithm updates the basis inverse (product form) by the use of an eta vector file.[1] Figure 9 illustrates the general logic of the SMPLEX routine. Much of the eta file is kept in core, if it gets too large for in-core operation, it is written on the binary file TAPE14. Each iteration the eta file is read to calculate the proper row to leave the basis and a new eta vector is computed and written on the file. In the maximization mode the "cost" of a unit flow is unit; thus the coefficients are updated in this fashion. The objective function is then computed by

$$z = \sum_j c_j x_j$$

---

[1] See G. Hadley, Linear Programming, Addison-Wesley Publishing Co., Inc., Reading Mass., 1962, for a complete discussion of the revised simplex method.

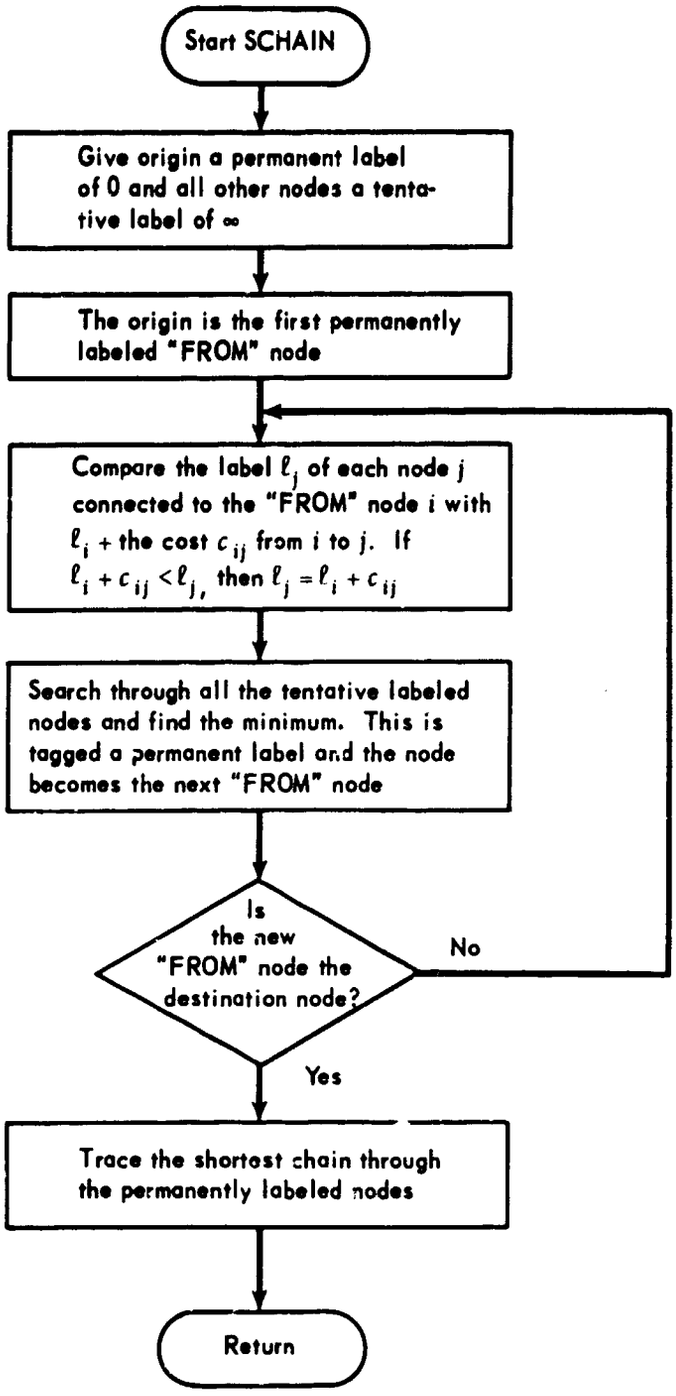


FIGURE 8. GENERAL FLOW CHART OF SCHAIN ROUTINE

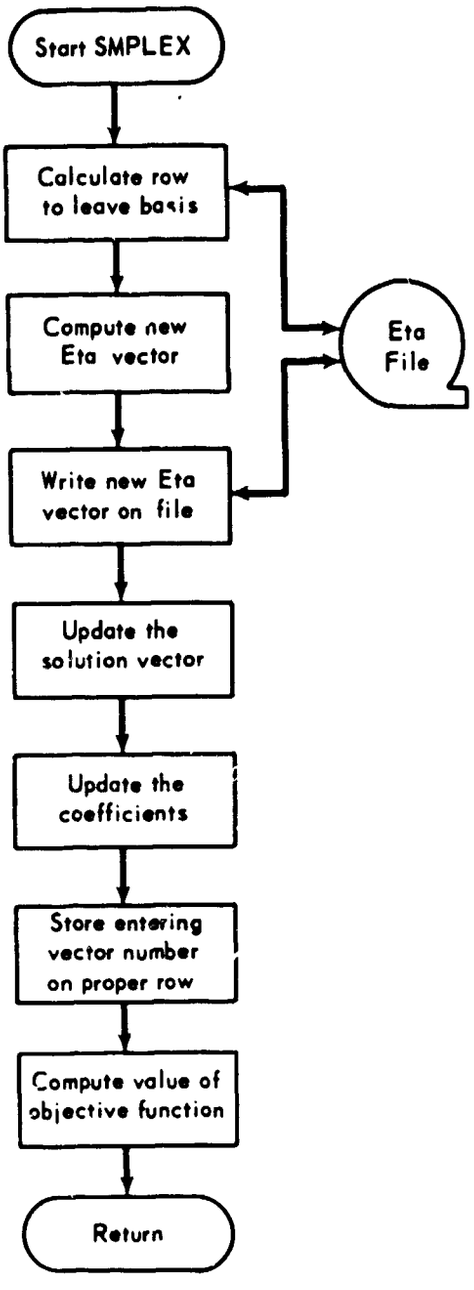


FIGURE 9. GENERAL FLOW CHART OF SMPLEX ROUTINE

where,

c_j = "cost" of vector j (unity if vector j is in the basis; zero if vector j is not)

x_j = flow on vector j

thus the value of z is the total flow; at optimum it is the maximum flow attainable.

(7) ALPHAS. This subroutine is used to generate the simplex multipliers each iteration after SMPLEX performs the basis update. Figure 10 illustrates the ALPHAS routine for the maximization objective. Note that if any simplex multiplier is found to be negative, a slack vector is created and immediately entered into the basis by the SMPLEX routine.

(8) WRITE. The WRITE routine is used at optimum to produce certain optimal data which are required for the output reports. As discussed earlier in this chapter under "Inputs and Outputs," these data are added to the input data file (TAPE1) when an optimal solution is reached. It also calls the routine ETAFLE to write the file used by the POP Module. Definitions and content of the data written on TAPE1 follow.

(a) Vector I.D. Numbers in Basis. NCHARC is the array of vector identification numbers present in the basis at optimization with their corresponding row numbers. As discussed under MAXIO, the vector numbers (NAJ) are assigned in sequential fashion beginning with NROW + 1 as they are entered into the basis during the algorithm. Since some vectors may leave the basis, only those present at optimization are relevant to the solution. The I.D. numbers permit the Output Module to identify these from the vector file (TAPE2)*.

(b) The Solution Vector. The solution vector, SOLVEC, is defined as $x_b = B^{-1}b$

where,

B^{-1} = Basis inverse

b = Right-hand-side vector.

Since the SMPLEX routine uses the product form of the inverse, the SOLVEC values are updated each iteration by an evaluation of the eta vectors. The SOLVEC values represent the variables in the basis; i.e.,

---

*See the section of this chapter on "Restart Mode" for a discussion of how these I.D. numbers also are used for restarting.

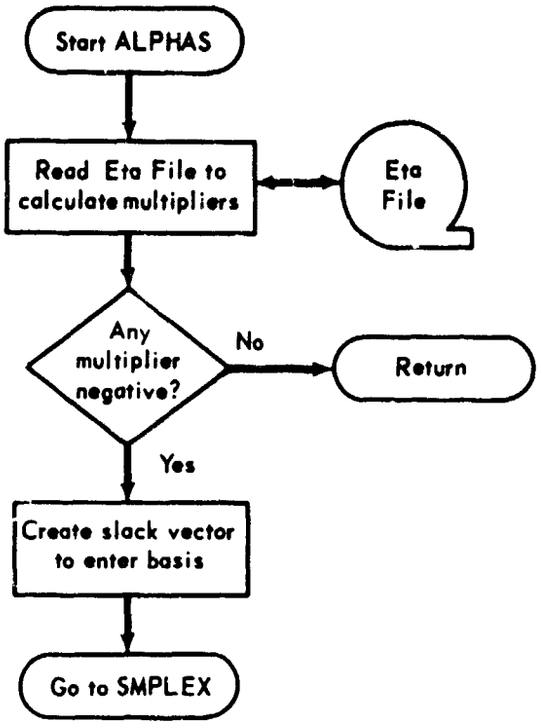


FIGURE 10. GENERAL FLOW CHART OF ALPHAS ROUTINE— MAXIMIZATION OBJECTIVE

the values of all active vectors are obtained from SOLVEC. The utilization of these values is discussed in Chapter 4, OUTPUT MODULE.

(c) The Simplex Multipliers (shadow prices). Simplex multipliers are computed each iteration by subroutine ALPHAS by reading the eta file. At optimum, these multipliers represent the shadow prices (or movement constraints) of each constraining row of the basis.

(d) The Value of the Objective Function. As discussed under subroutine SMPLEX in this section, the objective function is computed each iteration. At optimum it represents the maximum flow attainable in the maximization objective and it is reported in the Output Module.

(e) The Number of Iterations Required to Reach the Solution. The iteration count is sequential beginning with unity and is reported in the Output Module for information purposes. An iteration is defined as an update of the basis (the SMPLEX routine) required either in introducing a real vector or entering a slack vector into the basis.

c. Minimization Objective. Figure 11 is a simplified flow chart of RACAT min-cost algorithm (Phase I and II) similar in form to Figure 4 of the max-flow process. The problem as formulated in words is to find the set of routes and the allocation of resources which will satisfy fixed movement requirements at minimum total cost (or time). Delivery requirements are set for each commodity and the program determines the allocation which will meet these requirements at minimum cost. The min-cost solution is not independent of time. Since the quantity of a given resource used is a function of travel time, time is minimized for a given mode of transportation. However, in the min-time operating objective, routes are selected and resources assigned so as to meet the delivery requirements with minimum total flow unit-hours (e.g., ton-hours). The minimization objective of the RACAT Module solves either of these problems with the same algorithm; thus this discussion is applicable to both. Figure 12 is a general flow chart of the RACAT Module min-cost process showing the subroutine sequence for a normal run. The functions of each routine are discussed below.

(1) RACAT. The RACAT subroutine performs the same functions in the minimization problem as it does in a max-flow problem.

(2) SETUP. As in the maximization objective, SETUP performs initialization of the appropriate variables, vectors, and matrices and sets up for the operating objective. A minimization run also starts with an identity matrix as a beginning basis and a solution vector equal to the initial right hand side values. However, the number of rows for a minimization is set as

$$NROW = NARC + NRES + NCONST + NODP$$

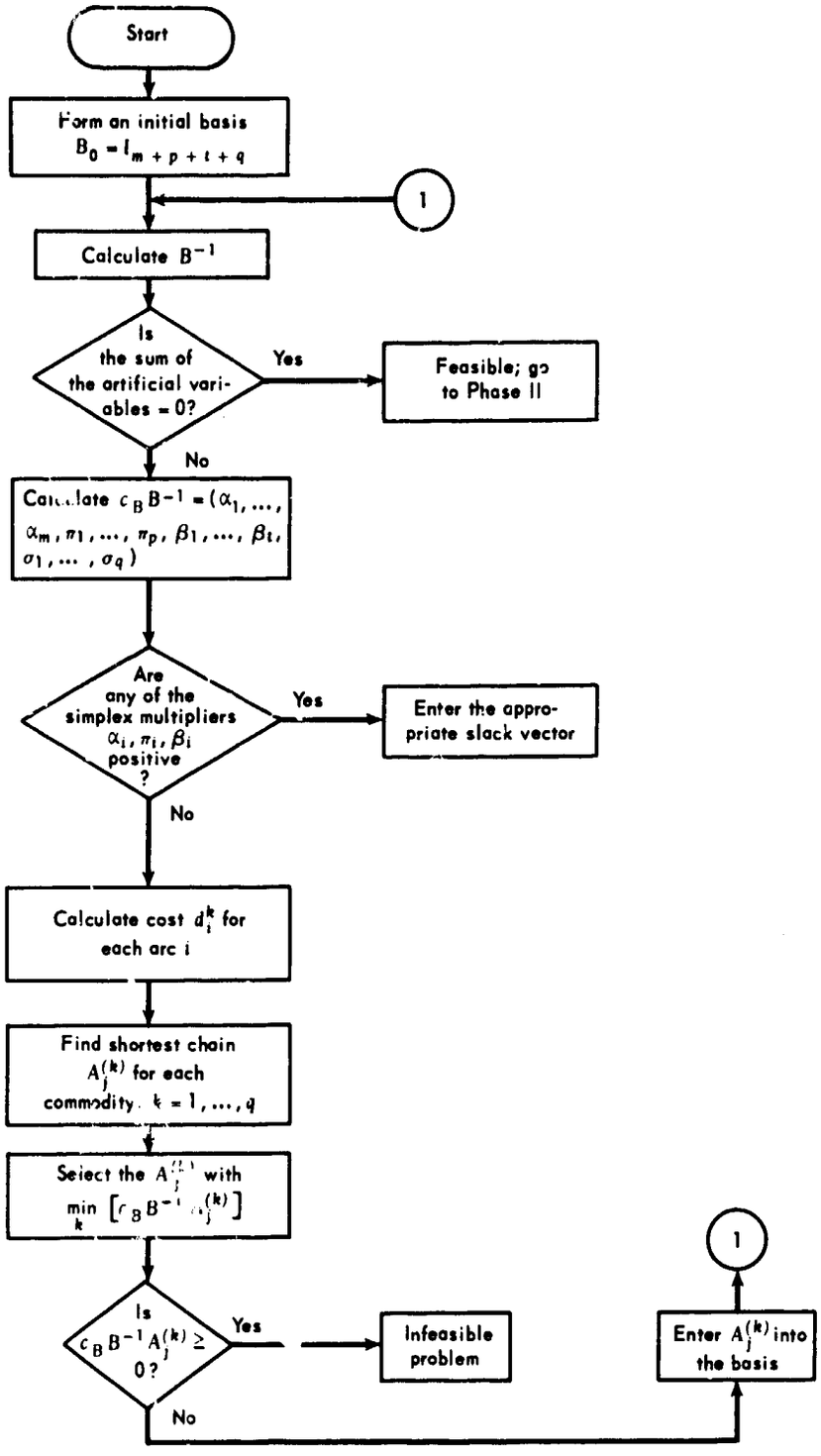


FIGURE 11. RACAT MIN-COST ALGORITHM FOR PHASE I
(Sheet 1 of 2)

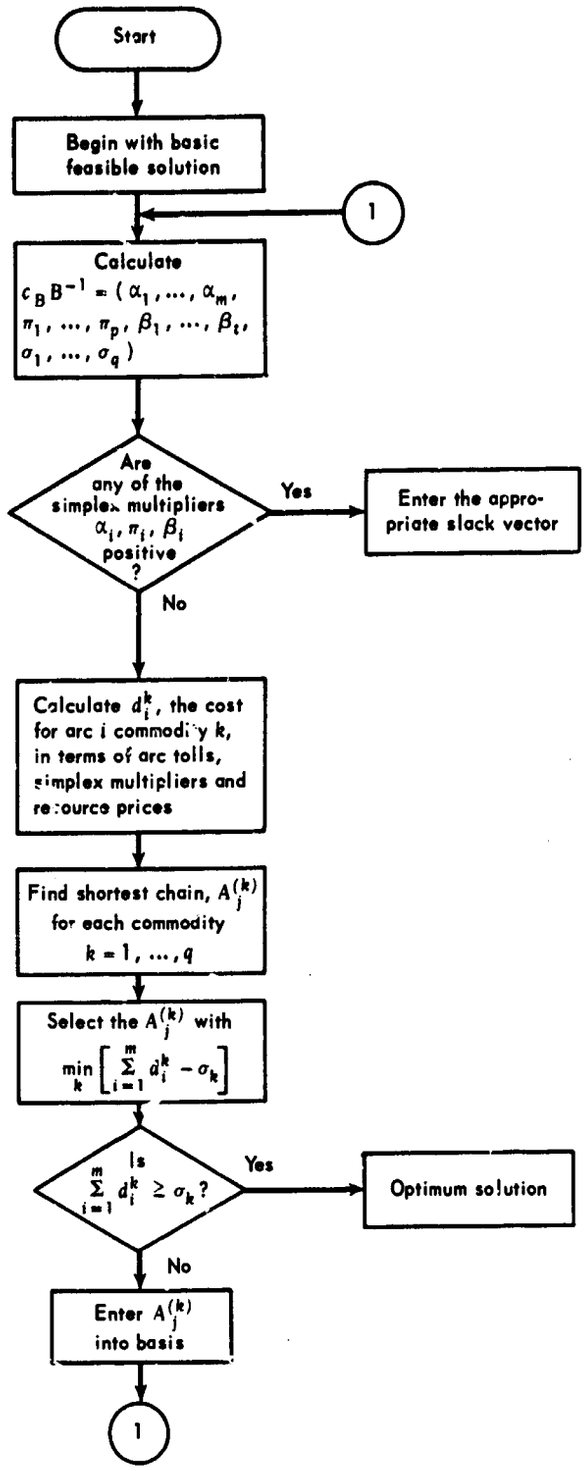


FIGURE 11. RACAT MIN-COST ALGORITHM FOR PHASE II
(Sheet 2 of 2)

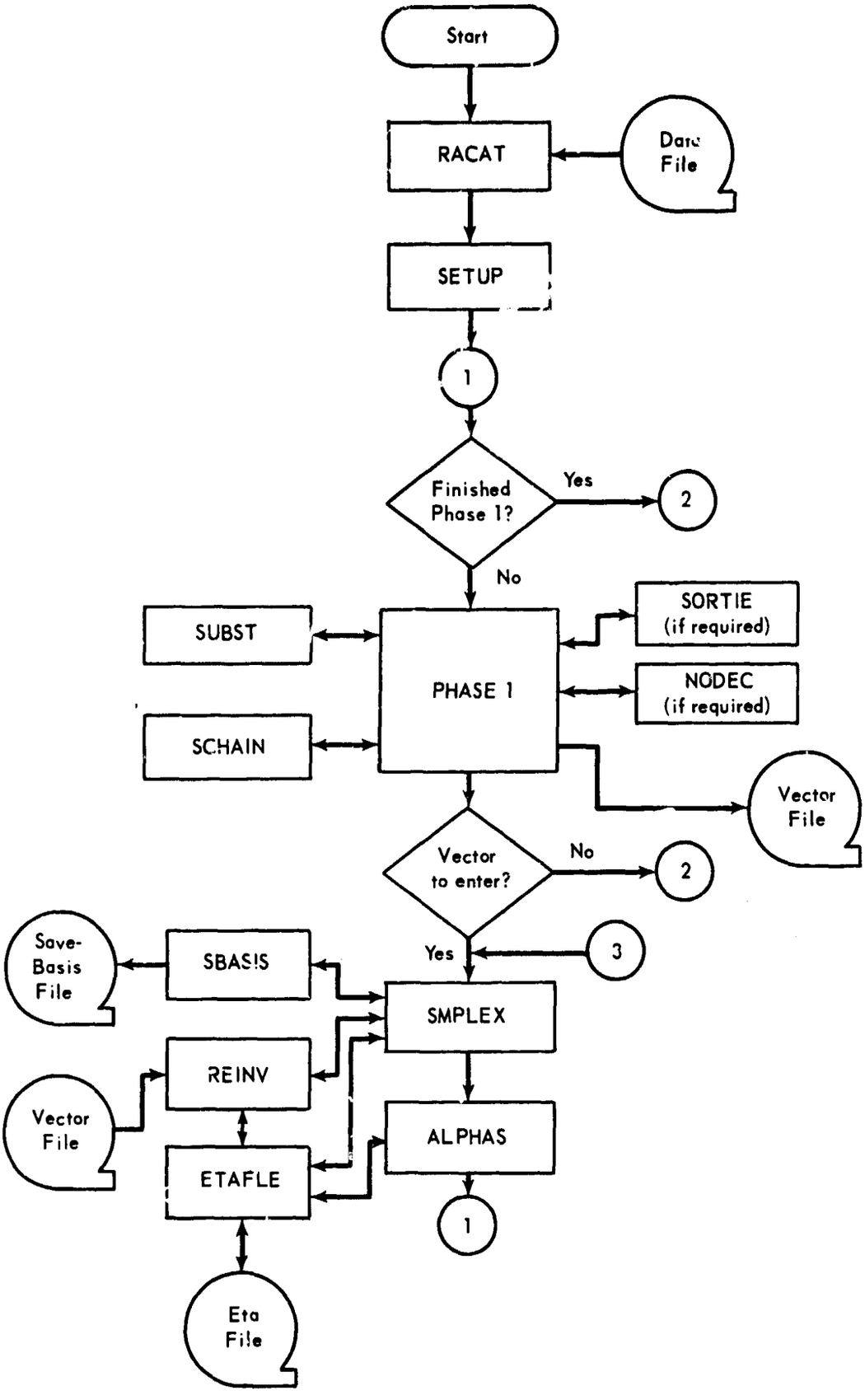


FIGURE 12. GENERAL FLOW CHART OF MIN-COST RACAT MODULE

(Sheet 1 of 2)

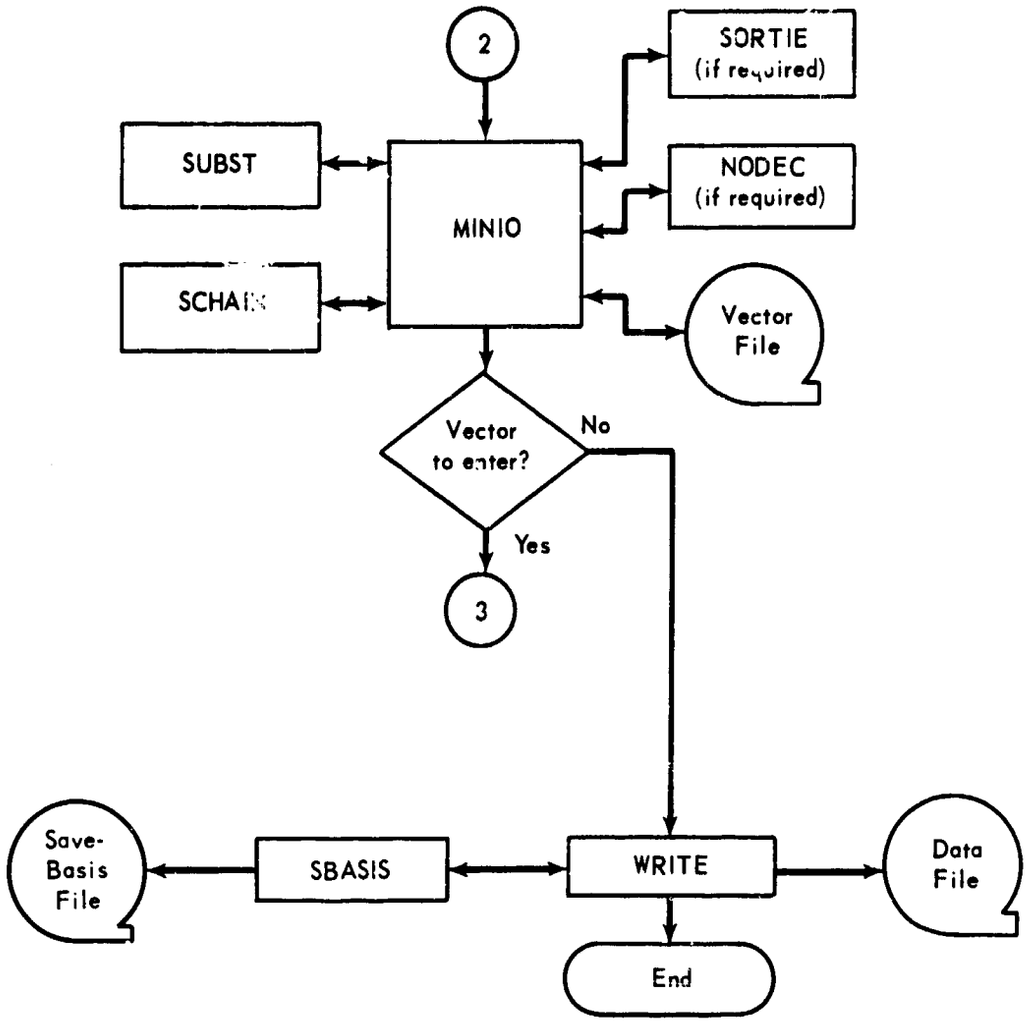


FIGURE 12. (continued)
(Sheet 2 of 2)

NARC = Number of arc constraints

NRES = Number of resource constraints

NCONST = Number of node constraints

NODP = Number of origin-destination movement requirements

(3) PHASE1. The PHASE1 routine operates as the controller for a minimization run in the Phase I operation, i.e., in an attempt to attain a feasible solution to the problem.*

(a) As such it performs similar functions as the MAXIO routine in the maximization objective and as MINIO, the Phase II controller to be discussed later. Figure 13 is a general flow-chart of PHASE1. The function of PHASE1 at each iteration is to:

1. Initialize variables.
2. Access an origin/destination pair (in sequence).
3. Call SUBST routine for appropriate resource vector.
4. Assign pseudocosts to arcs of the network.
5. Call SORTIE routine if V/TP constraints formulation is used.
6. Call NODEC routine if node constraints are present.
7. Call SCHAIN routine to generate a vector.
8. Evaluate vector to enter basis against simplex criterion.
9. If the vector does not improve the objective function, go to step 2.
10. Write this vector on the vector file.
11. Call SMPLEX routine to update basis.
12. If no vector meets the criterion, terminate RACAT run.

*See Enclosure 1, 2, and 3 for a mathematical discussion of the Phase I operation.

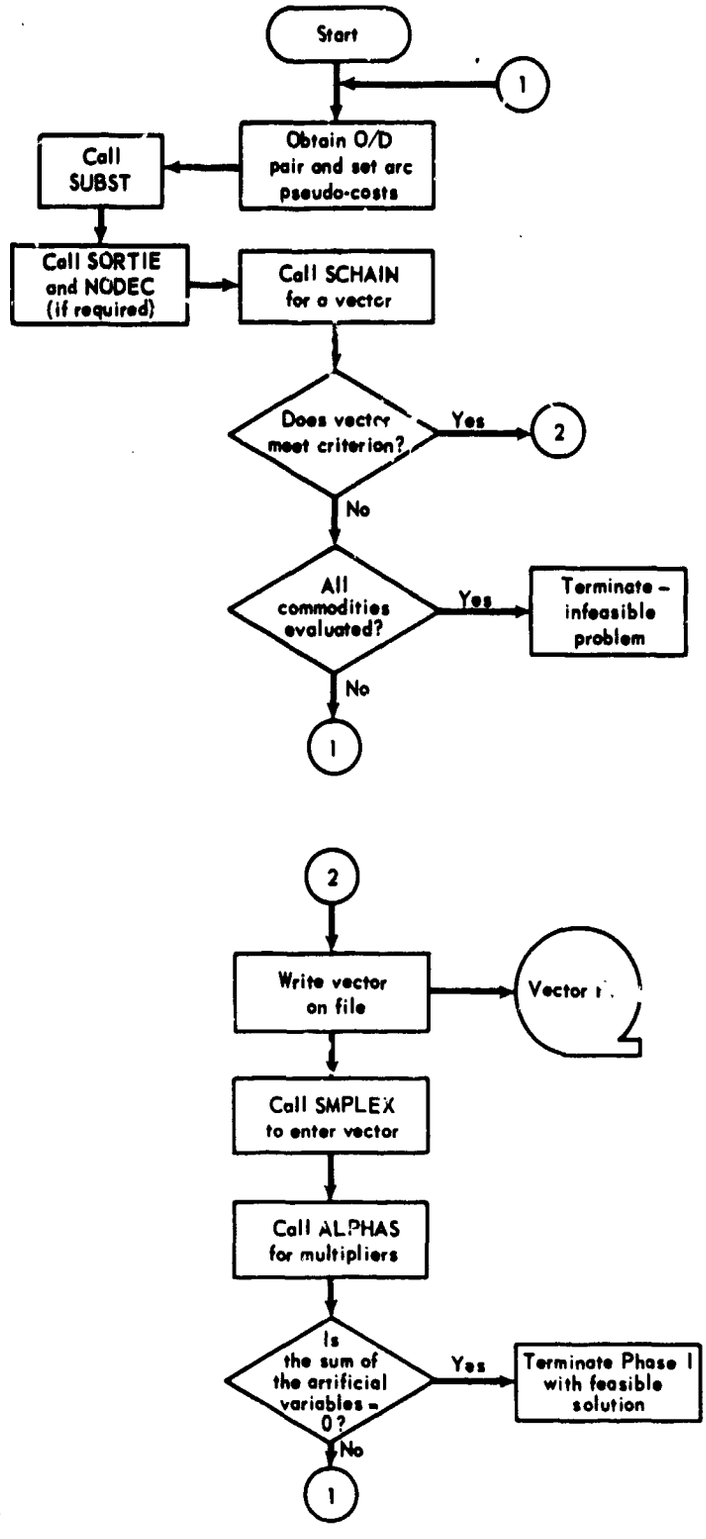


FIGURE 13. GENERAL FLOW CHART OF PHASEI ROUTINE

13. If the value of the objective function is zero (i.e., the sum of the artificial variables is 0) terminate PHASE1 with a feasible solution. Otherwise go to step 2.

(b) Each vector file record in the minimization mode is written as follows:

NAJ	=	Number of vector. A sequential numbering scheme beginning with NROW + 1.
NP	=	Number of the origin/destination pair.
NR	=	Number of the row in the basis in which the O/D intersect occurs. This row number identifies the O/D row to which this vector applies.
CTME	=	The delay time for this chain.
NAR	=	Number of arcs contained in the chain.
(AJ(I), I=1, NAR)	=	Arc row numbers in the chain.
(AJ2(I), I=1, NAR)	=	Vehicles payload factor for the arc whose row number is in the chain.
(COLUMN(I), I = NARC + 1, NODROW)	=	Resource coefficients and constrained nodes for this chain.

(c) As indicated in the PHASE1 flow chart (Figure 13) a feasible solution is attained when the value of the objective function reaches zero. This means that the delivery requirements have been met in a feasible fashion and Phase I is completed. If a feasible vector cannot be generated at any iteration to enter the basis, then an infeasible problem has been encountered and the RACAT Module terminates.

(4) SUBST. Operation of the SUBST routine for the minimization objective is similar to the description given earlier under the maximization objective. The criteria used for selection of the "best" substitution alternative is the only difference and is explained in Enclosure 2.

(5) SCHAIN. See the SCHAIN discussion under "Maximization Objective" which is also applicable here.

(6) SMPLEX. The discussion in "Maximization Objective" of the generalized SMPLEX linear programming code based on the product form of the inverse is also applicable to the minimization run with a few exceptions. Since the purpose of Phase I is to either remove the artificial variables (requirements) from the basis or to have them in the basis at a zero level, costs (c_j) of all structural and slack variables are set to zero and costs of unity assigned to each artificial. In Phase II the actual cost (c_j) is assigned each legitimate vector and is computed each iteration. Thus the final value of the objective function may be computed by:

$$z = \sum_j c_j x_j$$

as before. The value of z in a minimization run is the total cost of satisfying the delivery requirements.

(7) ALPHAS. The ALPHAS routine performs the same operations in computing the simplex multipliers as in the "Maximization Objective." However, if any simplex multiplier on a link row or on a resource row is found to be positive, the corresponding slack vector is created and immediately entered into the basis via the SMPLEX routine.

(8) MINIO. A transition operation is required in proceeding from a Phase I feasible solution to a Phase II processing. This involves a simple resetting of cost coefficients and control parameters. MINIO then assumes control for the Phase II solution process. It performs the same general functions as the PHASE1 subroutine in that it accesses O/D pairs and generates a vector to enter the basis. Each vector to enter is written on the vector file just as is done in Phase I. Phase II is terminated when the MINIO routine is unable to find a vector qualified to enter the basis. That is, there is no new route that will improve the solution. At this point an optimal solution for the minimization problem has been reached. Figure 14 illustrates the MINIO routine.

(9) ERROR. The ERROR routine is called when either of ten (10) major errors is discovered. The type of error is reported and the RACAT Module is terminated with a report of the active vectors present in the current basis. These errors are in general only discernible within RACAT and indicate a major problem in the proceeding of the algorithm. The types of errors checked for each iteration are as follows:

(a) The value of the objective function decreases (maximization objective).

(b) The value of the objective function increases (minimization objective).

(c) An infeasible vector enters the basis - indicates an unbounded problem.

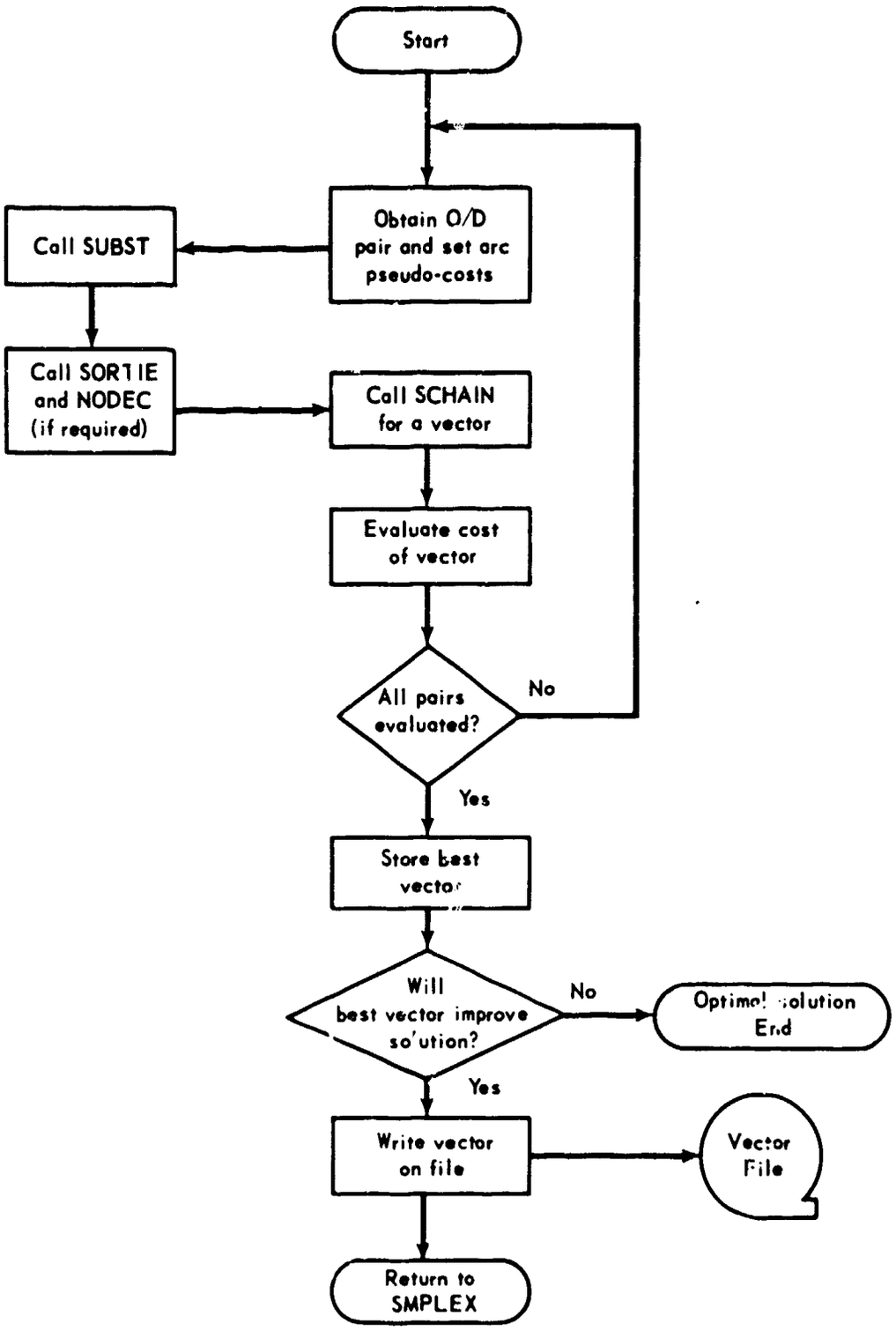


FIGURE 14. GFNERAL FLOW CHART OF MINIO ROUTINE

(d) A chain cannot be found between an O/D pair.

(e) An infeasible problem as stated (minimization objective).

(f) Algorithm has exceeded the maximum number of iterations as stated by the user in the input data. (see Chapter 2)

(g) Parity error on eta file TAPE14 (for CDC 6400).

(h) Tried to read backwards when TAPE14 at load point (for CDC 6400).

(i) The number of links adjacent to constrained nodes exceeds the maximum number of 700 as set by the variable NCLINK in the routine RACAT.

(j) The vehicle payload factor (or sortie constraint factor) is 0. This message may only occur when the V/TP formulation is used.

d. Inversion Mode. A technique used in standard linear programming codes is to "invert" the basis every "n" iterations in order to reduce the size of the eta vector file. This requires the elimination of all nonbasic vectors, i.e., those which entered the basis at some previous iteration and were replaced by a later vector. The "n" iterations at which inversion is to occur is specified in the input data file. It is felt that experimentation with this feature will provide the analyst knowledge as to the setting of this parameter. Some standard LP codes set it at 75 or compute the number as a function of significance. In ETNAM the analyst has the ability to specify it as a parameter. The formula for inversion time is simply

NNI = INVC + INNO

where,

INVC = The inversion iteration parameter

INNO = Iteration at which the inversion occurred and if the current iteration is equal to NNI then an inversion is called for.

Figure 15 is a general flow chart of the inversion mode. Subroutines SORTAJ and REINV are required to control this processing.

(1) SORTAJ. This routine controls the REINV process. First, the basic vectors are sorted using the generalized internal "Shell Sort" discussed in Chapter 2, INPUT MODULE. The array in which the basis vectors are stored is (NCHARC(I), I = 1, NROW) so that I

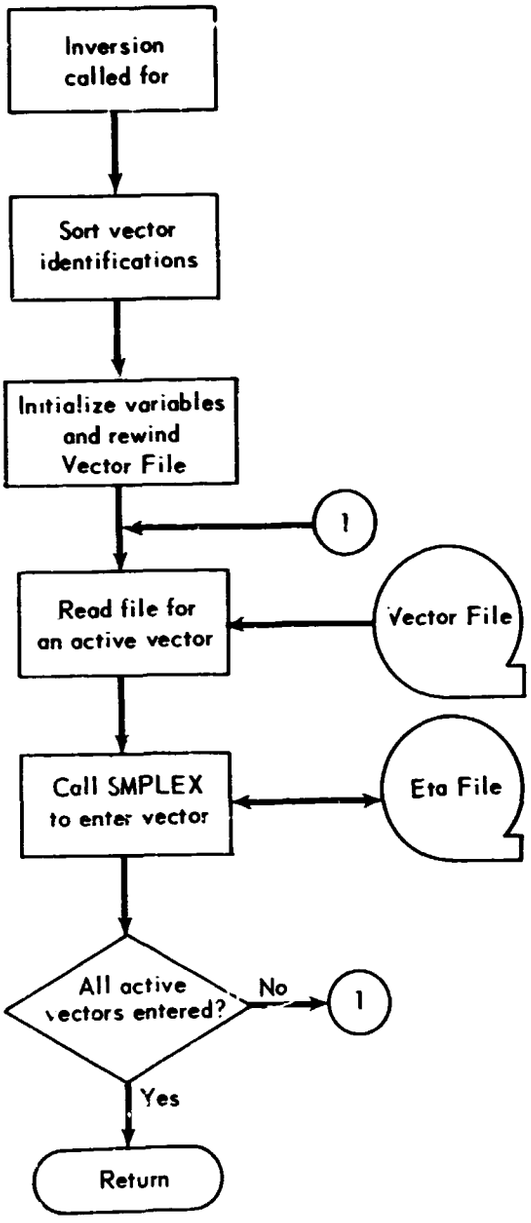


FIGURE 15. GENERAL FLOW CHART OF INVERSION MODE

corresponds to a row of the basis. The vector numbers in NCHARC may be classified into three categories. If NCHARC(I) = 0, then slack vector I is on row I (i.e., the slack vector I is said to be in its home position). If 0 < NCHARC(I) ≤ NROW, then the slack vector whose identifying number is given by NCHARC(I) has been entered the basis on row I. If NCHARC(I) > NROW, then NCHARC(I) gives the identifying number of a structural vector which has entered the basis on row I. A count is made of the number of structural vectors present in the basis at inversion time and this count is communicated to the REINV routine. Upon the completion of REINV, the SORTAJ routine resets for normal operation and returns to the routine from which it has been called -- either PHASE1 or SMPLEX.

(2) REINV. Figure 16 is a general flow chart of the REINV routine. The vector file is rewound to load point and the initialization of the basis and other variables is performed by subroutine SETUP. The vector file is then read to obtain a structural vector that is present in NCHARC, i.e., a basic structural vector number sorted in SORTAJ. Since the vector file has been written sequentially each time a vector entered the basis, the vectors may be located one at a time as the file is processed. When an active vector is found on the file, it is entered into the basis via the SMPLEX routine. The starting basis for the invert procedure is the all slack basis. The basis update is performed in a similar manner as a normal iteration in the maximization or the minimization operating objective. The difference is that the vector entering the basis is allowed to replace only a slack vector which was not in the basis at the time the invert procedure was initiated. A new eta file is generated representing only those structural vectors. Since the simplex multipliers are not required to generate a new vector to enter, the eta file does not have to be read backward and the shortest chain algorithm does not have to be used to generate a chain or vector. The inversion process is, therefore, a relatively rapid operation.

e. Restart Mode. The ability to restart a RACAT operation has many advantages. Two important uses occur when: a previous run has been terminated before reaching an optimal solution (e.g., when a time limit was set or a maximum number of iterations reached); or when it is desired to make certain small changes to the problem and obtain a new solution. As long as the number and types of constraint rows have not been altered, the restart mode may be used to "crash" in previously used vectors without computing simplex multipliers and generating a new vector to enter the basis each iteration.

(1) Figure 17 is a general flow chart of the RESTART routine. As mentioned in the discussion on "Input Data Requirements" in this chapter, one input card is required for the operation of the RACAT Module. A card with "NORMAL" punched in cols. 1-6 causes the RESTART routine to be ignored and a normal solution process is used. The RESTART operation is called by punching the word "RESTART" in cols. 1-7.

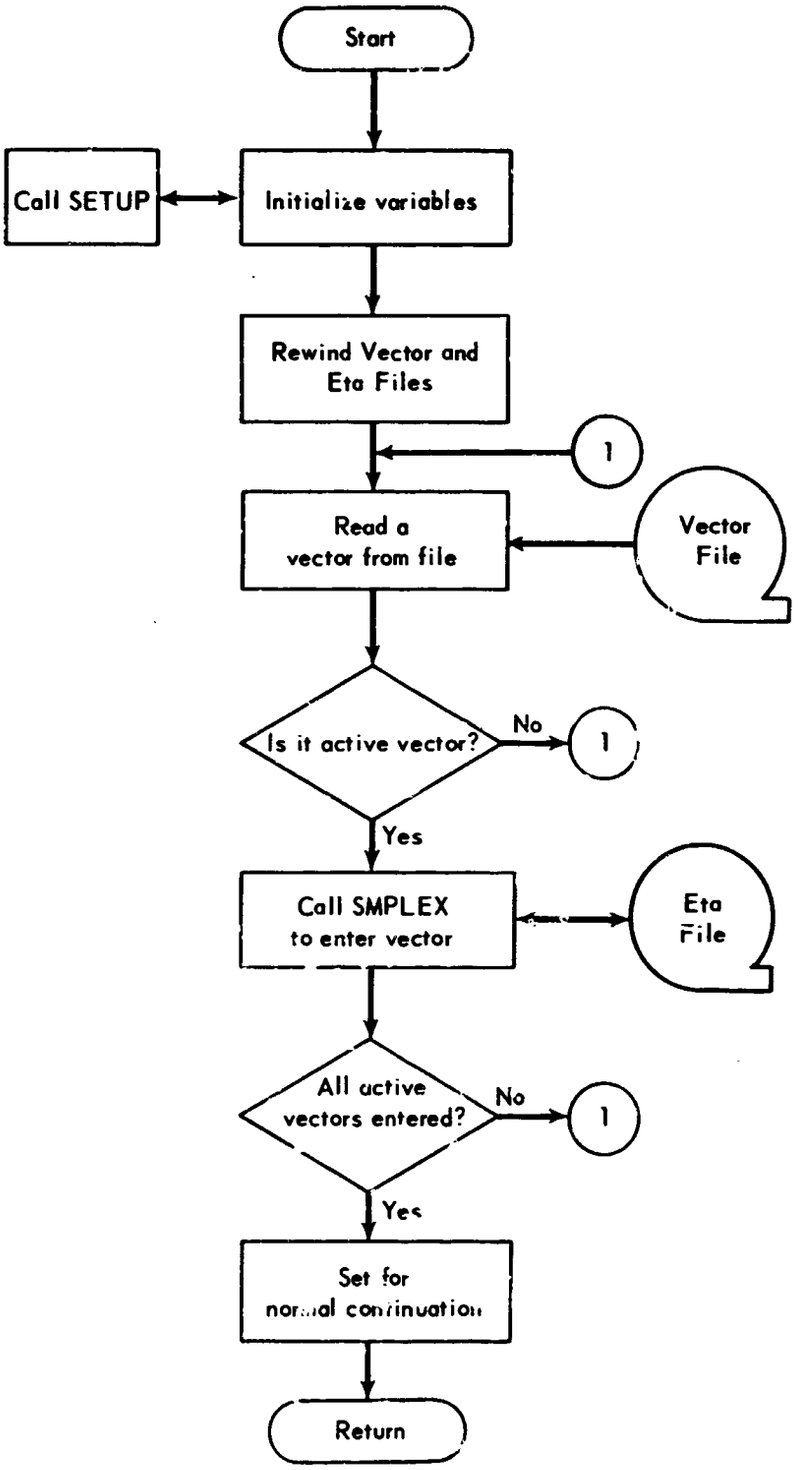


FIGURE 16. GENERAL FLOW CHART OF REINV ROUTINE

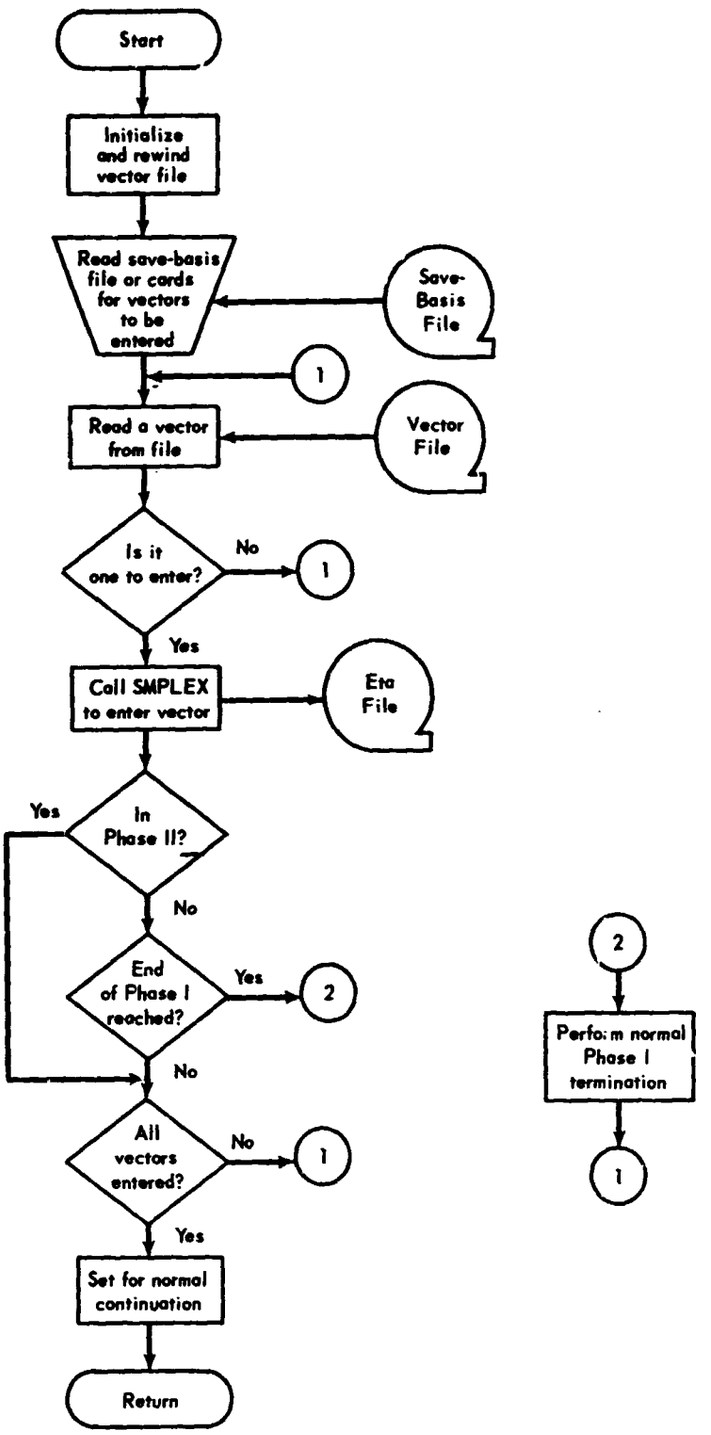


FIGURE 17. GENERAL FLOW CHART OF RESTART ROUTINE

(2) It is desirable to restart a job as near to the cutoff point as possible. The save-basis feature of ETNAM saves a checkpoint at desired iteration intervals. The frequency of saving a basis is given by the variable ITSAVE described earlier in this chapter. The saved-basis consists of the numbers of the structural vectors (or routes) that are in the current solution and a bit map, NVTX, which indicates which slack vectors are in the solution. This is called marking the slacks.

(3) Restarting may be accomplished in two ways: by marking the slacks and without marking the slacks. Whenever possible, "marking the slacks" should be chosen since it allows the program to continue from the point where the last basis was saved. Method 2 is less efficient and should only be used when it is not possible to use Method 1, i.e., when the save-basis file, TAPE9, is not available or when a change in the data has been made.

(4) To restart a run the following files are needed:

(a) TAPE1 - the input file from the run to be restarted. If some solution information has been written on TAPE1, it should be regenerated by the Input Module.

(b) TAPE2 - the edited version (from the VECTRS program) of TAPE2, the vector file, of the run to be restarted.

(c) TAPE9 - copied from TAPE (the save-basis file of the run to be restarted) by the VECTRS program.

(d) TAPE14 - a scratch file for the eta file. The eta file is packed into core. When it is too large to reside completely in core, it is written on TAPE14. For the CDC 6400 computer this must be a tape file.

(5) The card data to be input is as follows: RESTART is punched in cc 1-7 of the RACAT data card. The other parameters on this card are as described earlier in this chapter. However, a further word of clarification is needed for the parameter IREST. When IREST = 2, the vector numbers and the bit map, NVTX, must be input on cards immediately following the RACAT data card. The vector numbers are read in the FORMAT (I10,14I5/(16I5)). The integer in the first 10 columns of the first card is the count of the vector numbers to be read in.

(6) The bit map is stored in the first 20 locations of NVTX. For the CDC 6400, NVTX (1) has the bits numbered 60, 59, ..., 3, 2, 1, counting from right to left corresponding to the first 60 slack vectors, NVTX (2) has the bits numbered 120, 119, ..., 63, 62, 61, etc. NVTX is printed in OCTAL format at invert time, at the end of Phase I and at the end of a run. The vector numbers are printed when NVTX is printed and also at those iterations where the basis is saved. NVTX is read in the

OCTAL format (4020) with five cards being required. Those words of NVTX (1) to NVTX (20) that are not needed by the bit map, have zeros punched on the data cards. For the IBM System 360, the bits are numbered from left to right in NVTX and NVTX is printed in hexa-decimal format.

(7) A separate small program deck called VECTRS is used to edit the vector file and copy the save-basis file, of a run which is to be restarted. This procedure is optional but recommended since a restart will alter the contents of TAPE2 and TAPE9. For a VECTRS computer run, the old TAPE2 (vector file) and TAPE9 (save-basis file) are input as TAPE3 and TAPE8 respectively. Two new files referred to as TAPE2 and TAPE9 are generated. These become the TAPE2 and TAPE9 needed for the restart run. There is one parameter data card.

cc 1-5, NRES	=	Number of resources
cc 6-10, NCONST	=	Number of node constraints
cc 11-15, ICARD	=	0, Read number of basis vectors from TAPE8, i.e., old TAPE9. 1, Read numbers of basis vectors from cards following this data card, the format is (I10, 14I5/ (16I5)). The first 10 digit integer is the count of the vectors.
cc 16-20, ISORTI	=	1, if V/TP (sortie) constraints are used 0, otherwise

(8) A limit of 300 is set on the number of vectors that may be crashed into the basis. The vector identification numbers are stored in the array NOVEC.

(9) The restart operation is similar to the inversion process in that the vector file is read and the vector number checked to determin whether or not the vector is to be entered sequentially since the entire array NOVEC is searched for a match when a vector record is read from the file. If the vector is to enter it is structured properly and entered via the SMPLEX routine. During restart, just as in the inversion mode, the eta file is read and written but does not have to be read backward. This restart is also a relatively rapid process.

(10) RESTART checks each iteration in a minimization run to determine if a Phase I solution has been reached. If this occurs, the restart mode is temporarily interrupted and a Phase I update is performed. When the RESTART routine has completed "crashing in" all the specified vectors, control passes to the appropriate routine for a normal continuation toward a solution. In a maximization run, MAXIO takes control; in a minimization run, either PHASE1 or MINIO assumes operation depending on the phase in which the solution process is to be continued.

CHAPTER 4. OUTPUT MODULE

1. Purpose and Scope.

a. The purpose of the Output Module is to prepare and format output reports based on the results of the RACAT Module solution. The RACAT results are a series of mathematical data that are not easily interpreted in their raw form. It is the purpose of the Output Module to prepare and format these basic data so that they will be useful and convenient for the analyst.

b. The ETNAM Output Module has been designed as a data processing system that can be run automatically as a part of the overall ETNAM system, or can be run separately at the user's option. Two files contain the output data from the RACAT Module and are used as input to the Output Module. If these two files are saved, the Output Module may then be rerun. These same files are also used to restart the RACAT Module in the event the program is interrupted or in the event that a modification to the original problem is required.

c. The purpose of this chapter is to describe the Output Module, its inputs, outputs, and the methods used to prepare them. The formats of the ten reports that may be produced are included in the following.

2. Input Data Requirements. The Output Module has as its inputs the two files created by the RACAT Module, the data file and the vector file. The ETNAM system may be set up to run through the Input Module, the RACAT Module, and the Output Module in one computer session, or these modules may be run separately.

a. Data Files. The data file contains much of the data prepared and edited in the Input Module as well as the solution results from the RACAT Module. The vector file contains all the structural vectors that were entered into the "basis" since the start of the problem. Each structural vector is a unique mathematical description of a potential route from some origin to its related destination. This is, each vector explicitly describes the arcs making up a chain from an origin to a related destination, the resources required to move a unit of the commodity along that route, and identifies the constrained nodes. Vehicle payload factors, if the arc capacities are given in V/TP are also present. The RACAT algorithm is a process of bringing in such vectors in an iterative procedure until no better solution is possible. In the process of finding the solution some vectors, brought into the basis in earlier iterations, may be replaced by other vectors. The vectors which remain in the basis are referred to as active vectors. Those which have been replaced are inactive. All are on the vector file after a normal RACAT run (i.e., no restarts).

b. Data Arrays. The main program of the Output Module reads the data file and loads COMMON with the data it contains for use in the

preparation of the reports. The following is a description of the arrays in COMMON that are constructed by OUTGEN, the main program.

(1) MATARC (700,3). This array provides basic data for the arcs of the network. The columns of the array contain, respectively, the FROM node, the TO node, and the "mode" of the arc. It should be noted that arcs are not necessarily directed. That is, flow may occur that moves from the TO node to the FROM node. (See Chapter 2, Inputs.) The order in which these nodes are listed is the same as that used in the initial input form.

(2) RHS (900). The right-hand-side array contains the constraint values input to the Input Module. Each element of the array corresponds to a row of the basis. For example, the ith element of RHS (where i is equal to or less than the total number of arcs in the network) will contain the capacity of the ith arc.

(3) NORDES (150,4). This array contains the origin-destination pairs and information about them. The columns of the array contain the following information respectively: the commodity number, the origin node, the destination node, and the row number of the corresponding movement requirement. In the min-cost and the min-time modes of operation, movement requirements are established for each commodity. These requirements become constraint rows of the basis matrix and the fourth column of the NORDES array contains the row number corresponding movement requirement. In the min-cost and the min-time modes of operation, movement requirements are established for each commodity. These requirements become constraint rows of the basis matrix and the fourth column of the NORDES array contains the row number corresponding to the origin-destination pair. The origin-destination pairs and associated data are sequenced by commodity.

(4) NCD (20). This array is used for "bookkeeping" within the system. It indicates the row of NORDES containing the last origin-destination pair for the corresponding commodity. Up to twenty commodities may be handled within the ETNAM system. Thus the ith element of NCD is the row number in NORDES of the last entry for the ith commodity. The origin-destination pairs corresponding to the ith commodity are located from NORDES (NCD(i-1)+1) to NORDES (NCD(i)).

(5) SOLVEC (900). The solution vector array contains the values of the solution vector at the completion of the RACAT algorithm. Thus SOLVEC (j) contains the flow over the route represented by the activity vector occupying the jth column of the basis at solution time. The active vectors of the vector file thus represent routes that were included in the solution. The flow that was assigned to these routes may be determined by finding the location in the basis of these routes and referring to the corresponding element of SOLVEC.

(6) NCHARC (3600). This array contains cross-indexing information that permits the program to determine which vectors on the vector file are active vectors. As was mentioned above, the RACAT algorithm is an iterative procedure that selects route-resource combinations that will result in an optimum solution. Thus the vector file contains both active and inactive vectors. NCHARC contains an ordered set of active vector numbers and is used to "purge" the vector file of inactive vectors. The core storage occupied by NCHARC during the initial phase of the Output Module is later used for other purposes, hence the dimension of NCHARC is set to provide for its maximum use.

(7) TITLE (20). This array contains an eighty character title supplied by the user and accepted in the Input Module. This title is printed at the top of each page of the output reports.

(8) NAMOD (20). This array contains the user supplied names of the modes.

(9) NACOM (20). This array contains the user supplied names of the commodities.

(10) NARES (50). This array contains the user supplied names of the resources.

(11) Parameters. The parameters NROW, NRES, NARC, NMOD, NCOM, NCONST, and ISORTI are used here in the same sense as described in Chapters 2 and 3. There are other words and arrays in common, some of which are passed from the RACAT algorithm to the Output Module, that are not indicated in the above list. These words have highly specialized purposes having to do with the specific methods of programming the output reports.

c. Data Cards. In addition to the two files described above, one to six cards are read which indicate the report options to be followed in the Output Module. There are ten reports that may be prepared by the Output Module. Three of these are always prepared but the remaining seven may be suppressed at the user's option. The "Node Workload Report" summarizes the total flow by commodity that passes through selected nodes in the network. The user must specify those nodes in the network to be reported.

(1) Card Format 1. The first field (card columns (cc) 1-10) indicates the reports that the user does not want prepared. That is, if no reports are indicated on this card all reports will be prepared. A "1" punched in the ith column of the field indicates that the ith report is not to be prepared. (See Table 1 for a list of the reports and options.) The second field, cc 11-20, contains the number of the mode (right adjusted) to be reported in the Summary of Mode X Flow. The total number of nodes to be included in the Node Workload Report is punched (right adjusted) in the third field, cc 21-30.

TABLE 1. OUTPUT REPORT OPTIONS

1. Run Parameter
2. Summary of Network Flow; Ton-Miles Flow by Mode*
3. Summary of Mode X Flow - Optional
4. Movement Constraint Report - Optional
5. Route Report
6. Origin Workload Report - Optional
7. Destination Workload Report - Optional
8. Node Workload Report - Optional**
9. Summary of Commodity Flow - Optional
10. Summary of Resources Used - Optional

*Reported generated only when the arc capacities are expressed in tons.

**User specifies the nodes for which data is accumulated.

(2) Card Format 2. A second card format is used to input the node names of those nodes to be reported, as specified in the third field above. Ten node names per card are allowed with each name left adjusted within a field of 8 card columns. The number of cards used must be consistent with the total number of nodes as indicated in the first card type. The fields are cc 1-8, cc 9-16, ..., cc 73-80.

3. Outputs. As indicated previously, ten reports (Table 1) may be prepared using the Output Module. Some of these reports are so basic in their content that they are automatically prepared. Other, more specialized reports are prepared at the user's option. Since all these reports are prepared from the data and vector files, the Output Module may be run with alternate report options after the initial running of the full system. A brief explanation of each of the ten reports is given below. It is suggested that these explanations be read with the sample reports shown in the SD, Volume I, User's Manual, Enclosure 1.

a. The Run Parameter Report. The Run Parameter Report is a brief summary of the most basic parameters used in the run and the value of the objective function at solution. The number of substitutions, the number of modes, the number of commodities, the number of resources, the number of constrained nodes and the type of arc constraints used are shown. The number of iterations used by the RACAT Module in solving the network and the value of the objective function are also shown. The value of the objective function is the total flow achieved, the total cost required, or the total flow unit-hours (ton-hours if capacities are in tons) required depending on the run objective -- max-flow, min-cost, or min-time respectively. In addition, the report options requested by the user are presented.

b. Summary of Network Flow and Ton-Mile Flow by Mode. The Summary of Network Flow Report shows the total flow over each of the arcs in the network by commodity. Thus for any given arc in the network the

user can find the total flow by commodity over that arc that results from all chains using it. The total flow, the capacity, and the residual are shown. The flow by commodity is also detailed. Only those commodities that have a flow ≥.5 are shown in the section of the report labeled, "Flow by Commodity...". Those arcs with zero flow, that is those arcs that were not used in the solution, are listed separately. Only a total flow ≥.01 is shown.

(1) Two potential areas of misunderstanding should be noted. As has been mentioned previously, users have the option of making any of the arcs in the network "directed" or "undirected." That is, the user may specify whether flow is to be permitted in only the indicated direction or in both directions. If the arc is designated as an "undirected" arc the RACAT Module may assign flow to move from the TO node to the FROM node. Further, it is possible that a single arc may have flow in one direction for one commodity and in the opposite direction for another. This is quite normal in a multi-commodity problem. Thus it is not always possible to give precise meaning to the FROM and TO nodes of a link. In this report the FROM and TO nodes are listed exactly as they are designated on the input forms. The actual direction of any flow is easily determined from the chains shown in the Route Report.

(2) A second area of possible misunderstanding has to do with the commodity flows. The "Total Flow," "Capacity," and "Residual" are in equivalent tons or V/TP, while the flows by commodity are shown in their normal units (people, barrels, etc.). Thus the flows may not sum directly to the total flow. Additionally, the flows are shown as integers and may not exactly equal the total flow as indicated even when all flows are in tons because of rounding error.

(3) While the total flow by arcs is being accumulated, the flow by modes multiplied by the distance of the arc is also accumulated. The ton-mile flow and the percent of the total ton-mile flow for each mode are shown in the Summary of Ton-Mile Flow by Mode Report. This report is omitted if V/TP are used instead of tons for the capacities of the arcs.

c. Summary of Mode X Flow. This report is exactly like the Summary of Network Flow Report except that it shows only those arcs of the mode designated by the user. The basic idea behind this report is that the user may be especially interested in the flows over arcs of a particular mode. For example, analysts are often especially interested in the activity on transfer arcs. This report gives the user the ability to obtain a special report detailing a particular mode.

d. Movement Constraint Report. This report shows the factors that constrain the solution as found by the RACAT Module. For example, in a max-flow run certain arcs and certain resources may be the limiting

factors on the total flow achieved. The movement constraint report will list the arcs, resources, nodes and commodity movement requirements which are constraining and will show their corresponding shadow prices. The shadow price of a constraint row is defined as the change in the objective function per unit increase in the corresponding right-hand-side value, assuming that a change of basis is not required to maintain feasibility. In other words, it is the rate of change of the objective function with respect to the corresponding right-hand-side holding the other right-hand-side values fixed.

e. The Route Report. The Route Report is a detailed listing of all the chains used in the solution, together with the flow assigned to that route and the resources used to accomplish it. The node names of any saturated constrained nodes will also be printed. This report is the most basic report of the series and may often be the most useful. The report shows how the optimum flow meeting the requirements can be achieved through the actual selection of routes and the assignment of resources to that route. Only routes with flow ≥.01 are shown.

(1) Route Number. The route number is assigned sequentially simply for identification. Routes are listed in order by commodity. The route flow, cumulative flow, and travel time for the route are shown. The route nodes in order of their occurrence are then shown. This, of course, permits the user to trace the exact route used. The resources assigned to the route are shown for each route and for each commodity in total.

(2) Route Label. Each route is assigned a label which in most cases will be unique. If two or more runs are to be compared the label will tell which routes are duplicates even if the route numbers are different.

f. Origin Workload Report. The Origin Workload Report shows the flow by commodity out of each origin. Each node designated as an origin by the user in the input data is printed with its corresponding flow. If a node is both an origin and an intermediate node (on separate chains) only the originating flow will be reported. Flows are shown in their normal units (PAX, barrels, short tons, etc.).

g. Destination Workload Report. The Destination Workload Report is identical to the origin workload report except that the terminating nodes of chains are reported.

h. Node Workload Report. The Node Workload Report is identical in format to the Origin Workload Report and the Destination Workload Report. The user may request that up to fifty nodes be included in the list. In certain studies, particularly vulnerability studies, it may be useful to know the flow by commodity passing through a particular node.

i. Summary of Commodity Flow. The Summary of Commodity Flow Report summarizes the flow over the network by commodity. In min-cost and min-time runs these flows will correspond to the movement requirements. In max-flow runs these flows will be the maximum flow achieved, given the constraints of the network and the available resources.

j. Summary of Resources Used. The Summary of Resources Used Report shows the total resources assigned to the network in order to accomplish the flow, the residual (unassigned) resources, and the routes to which these resources are assigned.

4. Methods.

a. The Output Module actually consists of a main program called OUTGEN and eight subroutines. OUTGEN reads the data file, TAPE1, and loads COMMON as was described earlier. OUTGEN then prepares the Run Parameter Report and calls each of the tabulating and report writing reports in their proper sequence. Figure 18 is a schematic diagram of the relationship between the main program and the subroutines and their corresponding reports.

b. The JOTS subroutine reads the vector file, TAPE2, purges that tape of inactive vectors, and tabulates arc flows by commodity for the use of the SUMRY and XTRANX subroutines. The tabulation of flows by arc by commodity requires a large data array in memory. The actual space required is equal to the number of commodities times the number of arcs in the network. Thus if the maximum values of these variables are used, a table of 14,000 cells would be required for the 700 arc mode. JOTS prepares the table and SUMRY and XTRANX use the data to prepare the Summary of Network Flows and the Summary of Mode X Flows Reports respectively. Flows by commodity are shown in the same units of measure as inpu (e.g., PAX, barrels, tons).

c. JOTS, SUMRY, and XTRANX are programmed to permit a variable data array. If enough core storage is available to accumulate all the data in one pass, these subroutines handle the problem in this manner. If the network size is such that multiple passes are required, each subroutine is set up to handle the data in this fashion. A flow chart of the JOTS subroutine is shown in Figure 19.

d. The SUMRY and XTRANX subroutines are straight-forward data processing routines that simply format the data and produce the reports. SUMRY prints the data as accumulated by JOTS. XTRANX scans the data prepared by the JOTS routine printing only those arcs of the designated mode.

e. MOVCON is a straight-forward data processing routine that formats and prints the movement constraints. The shadow prices are simply the simplex multipliers from the RACAT Module and are contained in the COMMON array ALPHA.

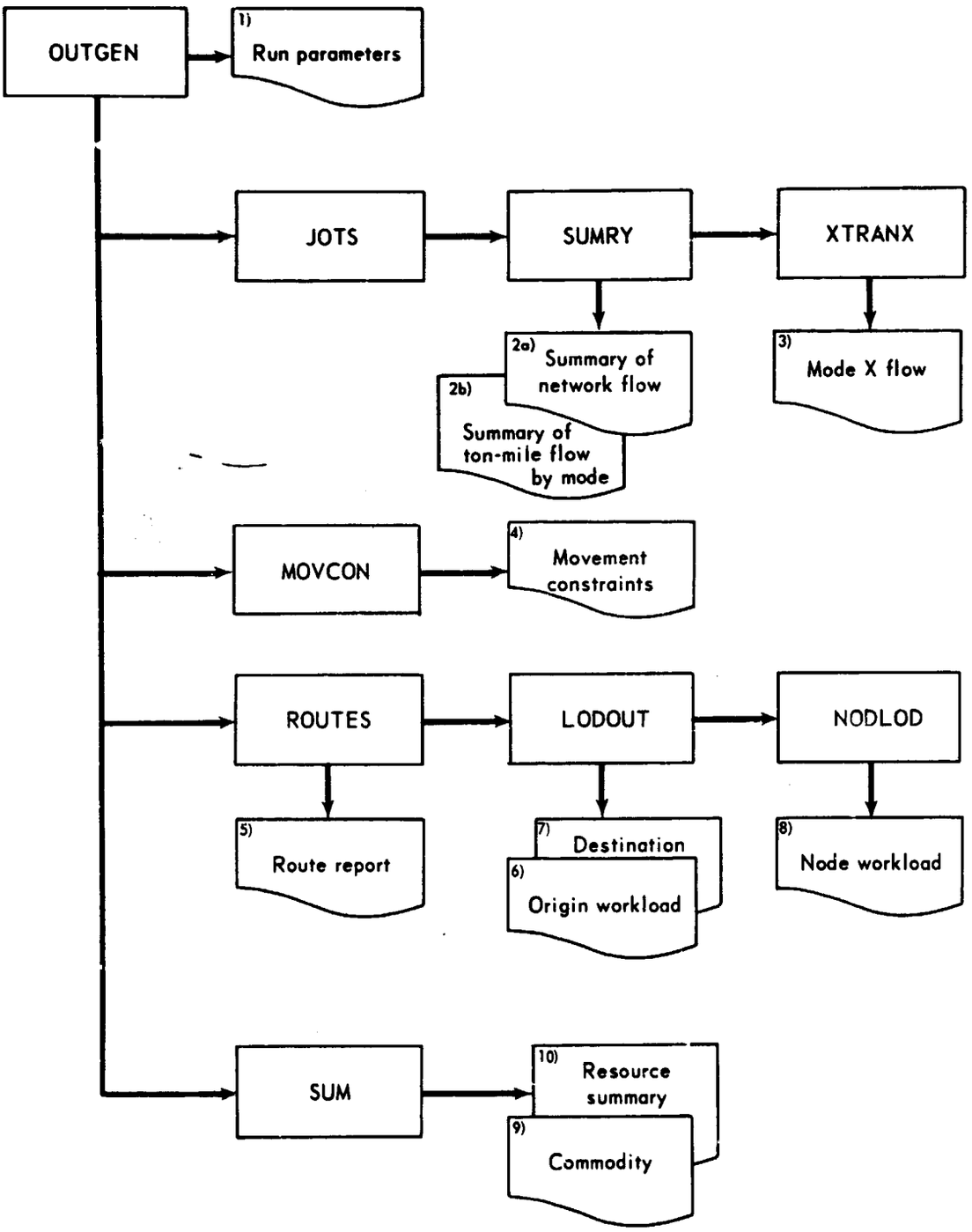


FIGURE 18. OUTPUT MODULE SCHEMATIC

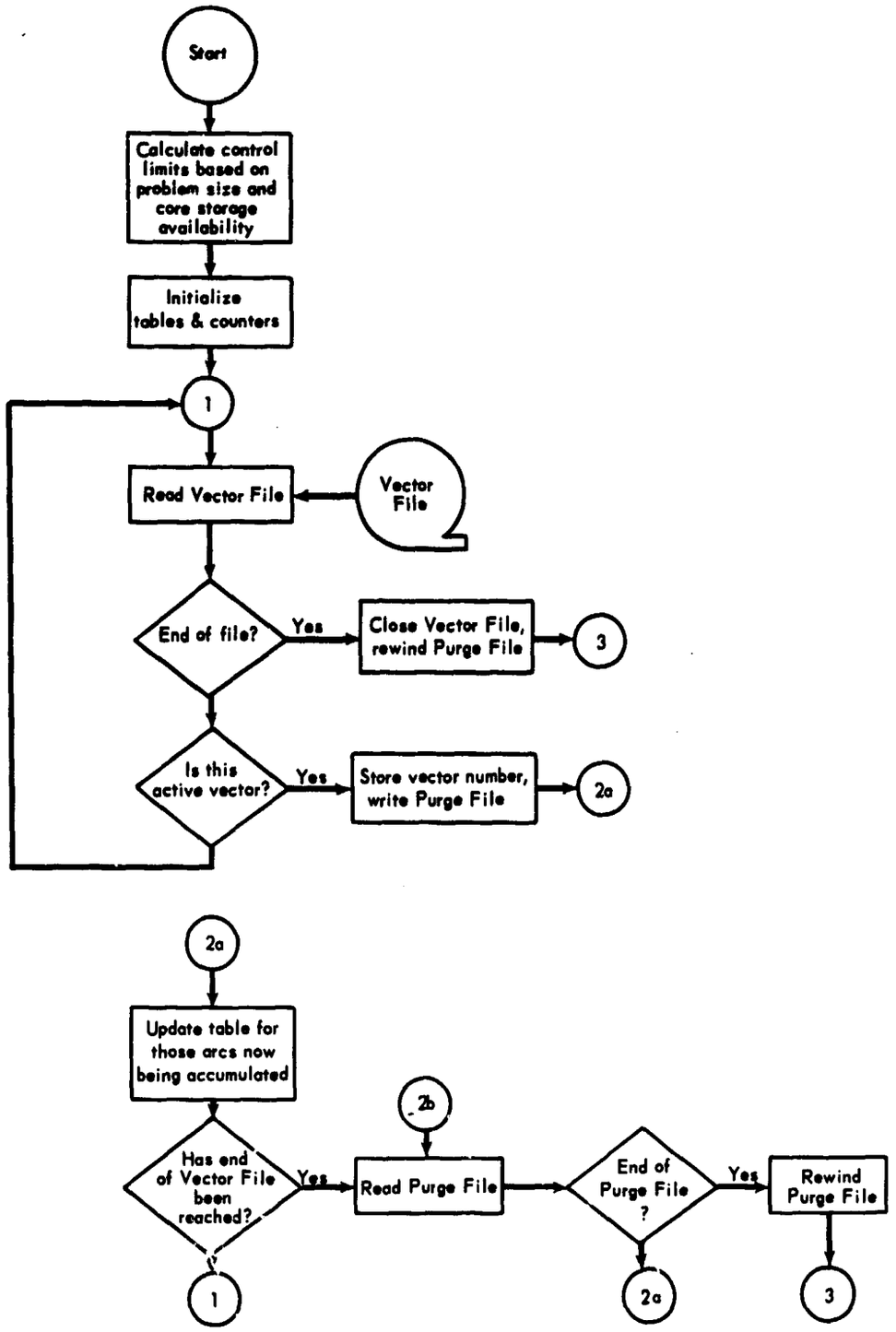


FIGURE 19. GENERAL FLOW CHART OF JOTS ROUTINE
(Sheet 1 of 2)

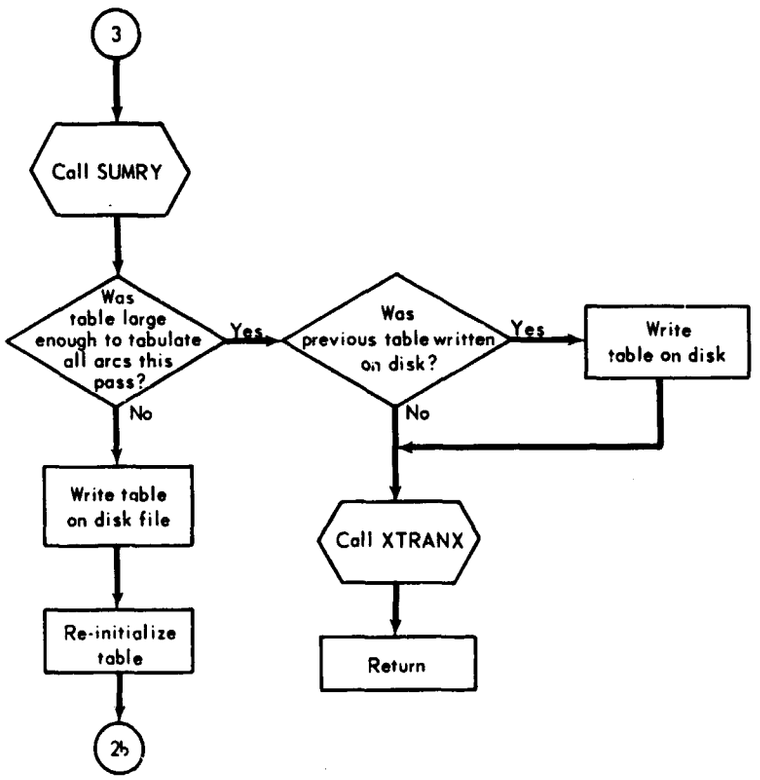


FIGURE 19. (continued)
(Sheet 2 of 2)

f. ROUTES is a somewhat more complex routine. Its basic information is obtained from the purged vector file which contains, in a raw state, all the data required. Routes must, however, determine the nodes contained in the chain and place them in their proper sequential order. The vector file record contains arc numbers and their order is not necessarily correct.

g. LODOUT and NODLOD prepare the Origin Workload Report, the Destination Workload Report, and the Node Workload Report. In preparing these reports it is essential that the data be carefully handled in order to avoid possible double counting. For example, an origin node may have traffic that goes through the node (coming from other origins) as well as traffic originating at the node. LODOUT and NODLOD handle this problem by examining and tabulating data from chains rather than from arcs. Thus both LODOUT and NODLOD subroutines are called by the ROUTES subroutine when a completed chain has been constructed.

h. SUMM is a relatively straight-forward data processing subroutine that prints out the Summary of Commodity Flow and the Summary of Resources Used Reports.

CHAPTER 5. POST OPTIMAL PACKAGE (POP) MODULE

1. Purpose and Scope. The POP Module provides the analyst with a tool for sensitivity analysis of the right-hand-side values. This is known as right-hand-side (RHS) ranging. RHS ranging is the only post optimal procedure possible for the current structure of the ETNAM system. RHS ranging gives the range of values a RHS element may assume, other RHS elements being held fixed at their given value, for which the current solution remains optimal. If the RHS element were to move just beyond the range, one of the vectors in the basis would have to be replaced by a vector currently not in the basis for the solution to remain optimal. If such a vector to enter is not found, then the solution would be infeasible for the RHS element beyond that end point of its range. Enclosure 4 gives a mathematical treatment of the applicability or inapplicability of the various post optimal procedures to ETNAM.

2. Input Data Requirements. Two types of data are required to operate the POP Module: the eta file and some parameters and data arrays written on TAPE14 by RACAT, and card input.

a. TAPE14 Input. This is a binary file and must be a magnetic tape in the Control Data 6400 version since it is accessed by LTRIO—a Control Data system subroutine—which can only access magnetic tape drives. The contents of TAPE14 are given below. On the IBM System/360, this may be a disk file.

(1) The Eta File. In the RACAT Module, the eta buffers and file records, if any, are all written on TAPE14. This is the complete eta file from which the basis inverse is computed.

(2) Parameters. These are: the array TITLE, NARC, NRES, NCONST, KETA (the number of eta vectors), ISORTI, NCOM and the array CONFX. The definitions for them are given in Chapters 2 and 3.

(3) Right-Hand-Side. The RHS elements are given.

(4) NCHARC. This is the array identifying the vector locations in the basis.

(5) SOLVEC. This is the solution vector.

(6) NORDES. This is the data array relating to the origin-destination pairs.

b. Data Cards. These data cards are used to specify an operational parameter NROW and the selection of RHSs for which a range is desired. The RHS selections may be made by key words or by specifying the lower and upper limits of a range of row numbers. There may be as many of these pairs of limits as desired. The data cards are followed by an END card. Figure 20 gives a schematic of the data card structure.

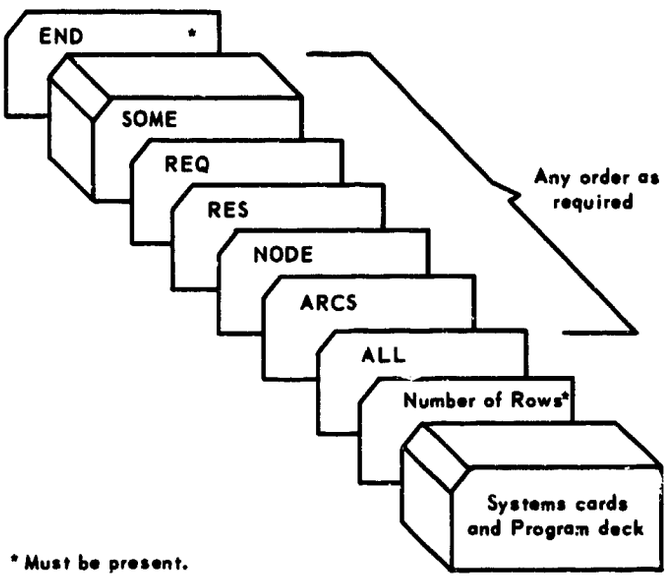


FIGURE 20. SCHEMATIC DESCRIPTION OF POP MODULE DECK

(1) Operational Parameter. This is the parameter NROW. Its value may be obtained from the first Input Module report. It is the number of rows in the problem. This value is punched as an integer, right adjusted, in card columns (cc) 1-5 on the first card. It is the only value on the first card and must be present for the eta file to be read correctly.

(2) Key Words. The key words are punched left adjusted in cc 1-4. Each key word is alone on a card unless it is the key word SOME, which is explained below. Each of the key word cards is optional. They may be in any order.

(a) ALL. This ranges each of the RHSs.

(b) ARCS. This ranges each of the RHSs associated with the arcs.

(c) NODE. This ranges each of the RHSs associated with the constrained nodes.

(d) RES. This ranges each of the RHSs associated with the resource inventories.

(e) REQ. This ranges each of the RHSs associated with the movement requirements.

(f) SOME. On each SOME card at least one set of limits must be specified. There may be up to seven pairs of limits on each card and as many cards as desired. If the lower limit of a given pair is n and the upper limit m then the RHSs corresponding to rows n through m are ranged, $n \leq m$. A SOME card, as has been noted, must have the word SOME in cc 1-4. Card Columns 11-20, 21-30, 31-40, 41-50, 51-60, 61-70, 71-80 are the fields for pairs of limits. A field may be left empty. The lower limit is right adjusted in the first five columns of a field (e.g., cc 11-15) and the upper limit is right adjusted in the second half of the field (e.g., cc 16-20). Other key word cards may follow a SOME card.

3. Output Reports. POP generates two reports.

a. The first report gives the title and identifies the row numbers associated with each type of constraint, i.e., arc, resource, node, and movement requirements.

b. The second report is a summary by row showing the RHS value; the total flow on arcs or constrained nodes, or the amount of resource used, or the commodity requirements moved (depending on the type of row); and the lower and upper value of that RHS range.

4. Methods. The Post Optimal Package consists of five routines, the main program POP and four subroutines, ETAS, RANGE, WRITE and ERROR. Each routine is discussed below and a block diagram is given in Figure 21.

a. POP. This is the main program of the Post Optimal Package. It controls the operational flow of the whole program. It calls ETAS to initialize the data arrays and the eta file. It reads the input data and checks for errors and calls ERROR if necessary. It calls RANGE to find the RHS range and it calls WRITE to generate the reports.

b. ETAS. ETAS is initially called by POP to read TAPE14 for the parameters, the eta file, the RHS vector and other necessary data as described in paragraph 2.

(1) $B^{-1}b$ is computed where b is the RHS vector and the basis inverse B^{-1} is a product of elementary column matrices κ_i'. The column of κ_i' which differs from the corresponding identity matrix is known as the eta (κ_i) vector. Only the non-zero elements of κ_i with proper identification are stored in the eta file. The identification consists of an index for each κ_i denoting the column of κ_i', and a bit map indicating the position in κ_i of the zero and non-zero elements for each eta vector i.

(2) $b = \text{col.}\ (b_1, b_2, \ldots, b_k, \ldots, b_m)$. If the range of b_k is to be found, the coefficient of b_k in the product $B^{-1}b$ needs to be found. This is done by separating b as $b = \text{col.}\ (b_1, b_2, \ldots, b_{k-1}, 0, b_{k+1}, \ldots, b_m) + \text{col.}\ (0, 0, \ldots, b_k, 0, \ldots, 0)$ and then replacing b_k by 1. Then the following product is computed $\kappa_q'\kappa_{q-1}' \cdots \kappa_2'\kappa_1'$ [col. $(b_1, b_2, \ldots, b_{k-1}, 0, b_{k+1}, \ldots, b_m)$ + col. $(0, 0, \ldots, 1, 0, \ldots, 0)$] where q is the number of eta vectors, m the number of rows and the 1 is in the k^{th} position of the last vector. This product must be ≥ 0.

c. RANGE. If there are m rows in the basis, the inequality $B^{-1}b \geq 0$ yields m separate inequalities. The range of b_k is essentially the set of numbers which satisfy the m inequalities. The upper limit of the range is the least upper bound of the limits given by the m inequalities and the lower limit of the range is the greatest lower bound of those m limits.

d. WRITE. This subroutine generates the reports discussed in paragraph 3.

e. ERROR. This subroutine gives error messages for errors generated by card input and by file input. The computer run is terminated where appropriate.

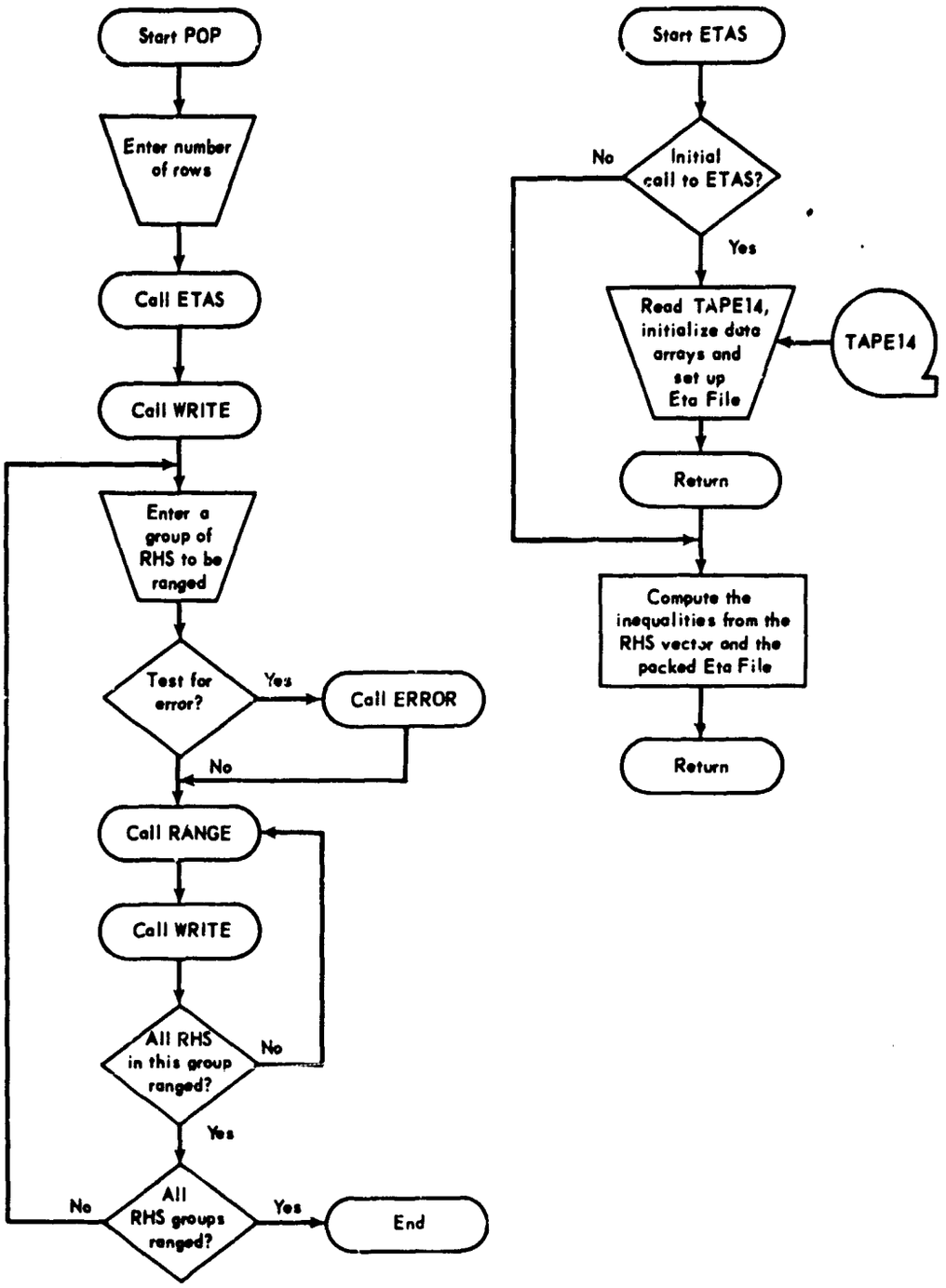
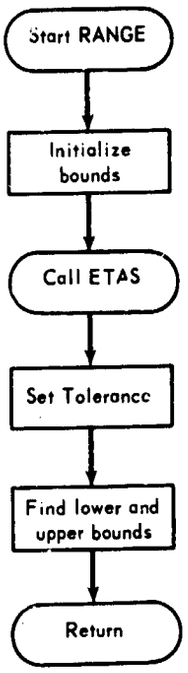
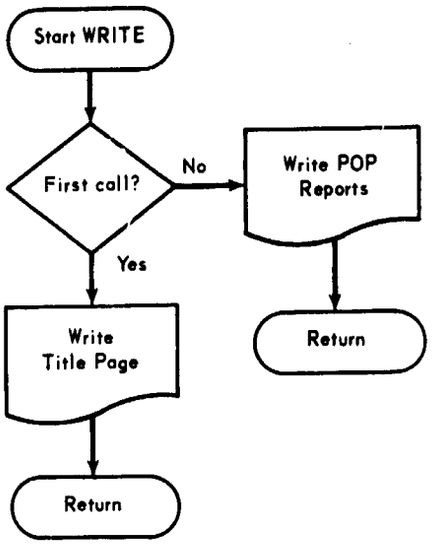


FIGURE 21. POP MODULE FUNCTIONAL FLOW CHART
(Sheet 1 of 2)





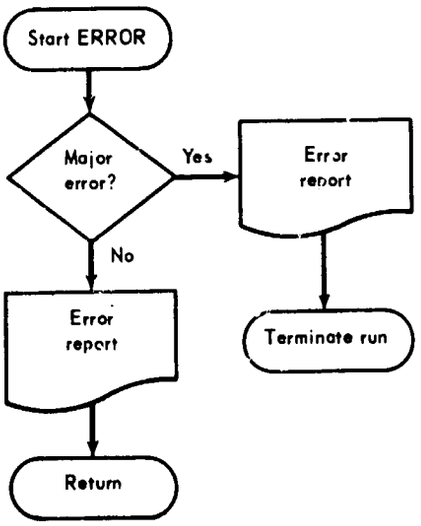


FIGURE 21. (continued)
(Sheet 2 of 2)

(1) File Input Error. When reading TAPE14, there are three error checks, each of which will terminate the run.

(a) If the number of rows of the problem (NROW) is not equal to the value specified on the card input, the run is terminated. NROW must be input correctly by card to read TAPE14 properly. But TAPE14 itself contains the parameter NROW that was used in the RACAT run. These two are compared and must be equal.

(b) If the number of eta records differ from those used in the RACAT run, the POP run is terminated. As TAPE14 is read for the eta file, the number of eta records is tabulated and compared with the parameter on TAPE14 which gives the number of eta records with which RACAT ended.

(c) The run is terminated if there is a parity error on TAPE14. If the parity error persists, the RACAT run may be restarted from the optimal basis and TAPE14 will be regenerated.

(2) Card Input Error. The computer run is terminated if a key word is misspelled. If on a SOME card a lower limit or the corresponding upper limit is zero, an error message is generated and that limit pair is not used. In this case the computer run is not terminated.

CHAPTER 6. ACCURACY

1. Purpose and Scope. Whenever a computer program—like the ETNAM computer system—is written which involves data inputs and many numerical computations, the analyst must always reckon with the possible inaccuracy of the results. This depends on the accuracy of the input data, the number of bits per computer word (which determines the significant digits in the computation), and the tolerance used in the program to decide when two numbers are considered equal. These aspects will be discussed in this chapter and an attempt made to draw the analysts' attention to areas most susceptible to numerical problems.

2. Inputs. The preparation of the input data consists of three distinct activities. Errors are easily made in all three of these activities.

a. Data Collection. Collection of the numerical data (like the resource prices, productivities, and inventories; the network link capacities and distances; and vehicle speeds) is a potential source of errors. The resource productivities are usually numbers in the range 10^{-3} to 10^{-8}. It is difficult to recognize errors in such numbers since they are not intuitively related to operating characteristics. Unfortunately even a small error in productivity data may result in a solution quite different from the optimum. For example, if a POL route using waterway in the optimum solution is closely competitive with a POL route using rail, even a small error in favor of rail will result in a solution different from the optimum. It is difficult to know in advance whether a particular resource will be sensitive. It is often possible to determine from the Movement Constraint Report which items of the input data are the most sensitive. Thus it is worthwhile to recheck these sensitive data after a solution.

b. Network Construction. A transportation and activity network consists of various kinds of links (e.g., a road between two cities could be a link in the network as could the transfer activity of offloading a ship onto trucks).

(1) Care must be taken in constructing the network to include all the necessary links. For a transportation network, separate networks are usually constructed for each transportation mode and are then "hooked" together with links representing transfer operations. The network for each transportation mode may be drawn on paper and checked quite carefully. The transfer links are more difficult to check since the FROM node is on the network of one transportation mode and the TO node is on another.

(2) Most links may be used in either direction, but to model the actual activity correctly it may be essential that certain links be directed. Consider Figure 22, for example.

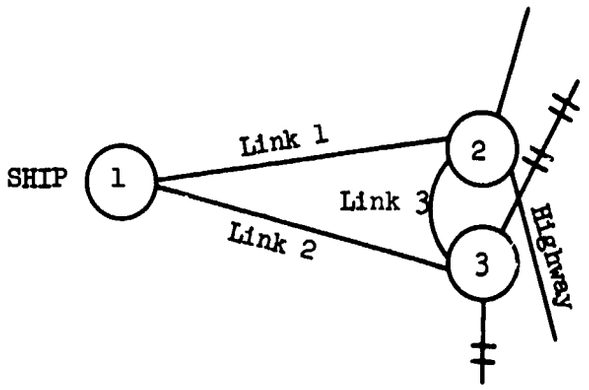


FIGURE 22. EXAMPLE OF NEED FOR DIRECTED LINKS

Node 1 is the origin, say a ship, of a commodity. Another commodity is already being shipped on highway to Node 2 where it is transferred to rail via Link 3, using it to capacity. Link 1 and Link 2 are transfer links from the ship to highway and railroad respectively. More of the second commodity could be transferred from Node 2 to 3 than the capacity of Link 3 by moving from highway to ship to rail. A "nonsense" route could thus be created. In order to prevent this, Links 1 and 2 should be directed from Node 1 to 2 and from Node 1 to 3 respectively.

(3) In structuring a network it is possible to overconstrain it. That is, not enough throughput capacity is available to make all the delivery requirements. As a result, an infeasible solution ensues. This happens frequently on newly structured networks. To circumvent the problem, a dummy link is inserted into the network for each origin-destination pair. But a very high toll is assigned to it. This means that the algorithm will move as much of the requirements as possible over the normal network links and use the dummy or escape links only as a last resort. In this way the analyst gets an optimal solution and is also told how much of the delivery requirement was not met.

c. Preparation of the Data in Punched Card Form. Clerical and keypunch errors can result in wasted computer time. To facilitate the data checking, the recommended procedure is to run the Input Module first and study the generated output reports of the input data. If the network is a newly structured one, it may take the first RACAT run to fully debug it.

3. Calculations. When doing numerical computations, the problems associated with truncation and tolerance—usually referred to as numerical difficulties—must be considered. The sixty bit word of the Control Data 6400 and double precision on the IBM System/360 reduce

this problem, but loss of significance may still occur. The ETNAM computer programs have been written incorporating a number of features designed to reduce the effect of the numerical problems.

a. Re-inversion. An example of such a feature is the re-inversion technique. The inverse of the basis matrix is used repeatedly in the algorithm and must therefore be available to the program. Since it is too large to be maintained in core storage, the basis inverse is kept in the "product form" on the eta file. Each iteration of the algorithm generates an "eta vector" and the product of all the eta vectors on the file is the basis inverse.

(1) After 100 iterations there are 100 eta vectors on the file and 100 vector multiplications must be performed to generate the data necessary for the next iteration. This is time consuming and it increases the likelihood of numerical difficulties because of the repetitive multiplication. It is typical of these problems that routes (vectors) entered into the basis in the early stages may not remain through the later stages. Thus it is quite probable that after 100 iterations, a far lesser number (e.g., 40) of the routes may actually be active. By a process called re-inversion we can reduce the length of the eta file to precisely the number of active routes in the current basis. Thus periodic re-inversion is usually worthwhile.

(2) There is a trade-off involved. A shorter eta file will, of course, reduce the time required per iteration, but re-inversion itself requires time. In the ETNAM system, the user may specify how many iterations are to be executed before re-inversion takes place.

b. "Tie" Routes. Routes consist of a sequence of connected links of the network and the resource required to sustain a flow over them. The desirability of including any route in the basis is determined by the sum of the costs of the links of the route. The cost of each link is in terms of its toll, the simplex multipliers (shadow prices) of the link and of the adjacent constrained nodes, and of the resources used on the link. Sometimes two or more routes may have the same value, which means that the routes are "tied" for selection as the "shortest" route. Ties are broken by choosing the route with the least distance. If a secondary test results in a tie too, then the first route found is used. "Tie" routes are not uncommon. In the maximum flow problem the objective is to maximize throughput without regard to time or cost and hence maximum capacity routes rather than minimum cost or time routes are chosen. These maximum capacity routes are normally driven out of the solution when resources are constrained. The shorter routes are chosen in order to effectively use the resources.

c. Tolerance. In the RACAT algorithm there are calculations which would result in a zero value if numerical accuracy were absolute. Since there are problems in numerical significance—and there would be no

matter how large the computer word—these calculations can result in some very small number rather than zero. For example, in the computation of simplex multipliers the program treats any simplex multiplier less than .00000001 as if it were zero. This value is known as the "tolerance" value. It eliminates any iteration that might result in only a tiny improvement or which might be caused by small errors due to numerical problems.

4. Output Processes. Each module of the ETNAM computer system has some output processes. The output consists of files and printed output. All files are written in binary rather than BCD; this precludes the need for converting the data to BCD and introducing round-off errors. In the printed output, the FORTRAN F format has been used whenever possible to print decimal numbers (e.g., 126.732) for ease in reading. If range of a number is large, to avoid overflow or lack of significance, the FORTRAN E format—the scientific notation—has been used (e.g., the number .000032 would be printed as 3.2 E-05, the E-05 standing for 10^{-5}).

REPRINT OF

RAC PUBLICATION OF RAC-P-44, OCTOBER 1968

A MODEL FOR OPTIMAL MULTICOMMODITY NETWORK FLOWS WITH RESOURCE ALLOCATION

LOGISTICS DEPARTMENT
PAPER RAC-P-44
Published October 1968

A Model for Optimal Multicommodity Network Flows with Resource Allocation

by
John E. Cremeans
Gene R. Tyndall

RESEARCH ANALYSIS CORPORATION

McLEAN, VIRGINIA

ACKNOWLEDGMENTS

The research leading to this paper was under the overall direction of BG I. L. Allen, USA (Ret) and was done under contract with the Defense Communications Agency in support of the Special Assistant for Strategic Mobility, Joint Chiefs of Staff.

The authors wish to acknowledge the encouragement and assistance given them by Mr. Donald Boyer of The George Washington University, Professor Mandell Bellmore of The Johns Hopkins University, and Dr. Robert A. Smith of Research Analysis Corporation.

CONTENTS

A Model for Optimal Multicommodity Network Flows with Resource Allocation

ABSTRACT

The problem of determining multicommodity flows over a capacitated network subject to resource constraints may be solved by linear programming. However, the number of arcs and resources in many applications is such that the standard formulation becomes very costly. This paper describes an approach—an extension of the column generation technique used in the multicommodity network flow problem—that simultaneously considers network chain selection and resource allocation, thus making the problem both manageable and optimal in the sense that flow attainable is constrained by resource availability and network capacity. Extension to the minimum-cost formulation is proposed and computational experience is discussed.

1. INTRODUCTION

The problem of multicommodity flows in capacitated networks has received considerable attention. Ford and Fulkerson[1] suggested a computational procedure to solve the general maximum-flow case. Tomlin[2] has extended the procedure to include the minimum-cost case. Jewell[3] has pointed out the strong historical and logical connection between this solution procedure for the multicommodity problem and the decomposition algorithm of Dantzig and Wolfe.[4]

A related problem, which has not been directly addressed, is the determination of multicommodity flows in a system constrained by resource availability. For example, flows in transportation networks are constrained by available resources that must be shared by two or more arcs in the network. The determination of the set of routes and the allocation of resources to these routes to maximize multicommodity flows or to minimize system cost in meeting fixed flow requirements can be applied to many problems in logistics and other areas. This paper discusses solution procedures for multicommodity network flows with resource constraints in the maximum-flow and minimum-cost cases.

2. THE MAXIMUM MULTICOMMODITY NETWORK FLOW PROBLEM

Consider the multimode, multicommodity network $G(N,\mathcal{A})$. N is the set of all the nodes of the network. $\mathcal{A}$ is the subset of all ordered pairs (x,y) of the elements of N that are arcs of the network. $\mathcal{A}_1, \ldots, \mathcal{A}_m$ is an enumeration of the arcs. Each arc has an associated capacity $b_{(x,y)} \geq 0$ and an associated cost (or distance) $d_{(x,y)} \geq 0$.

For each commodity k ($k = 1, \ldots, q$) there is a source s_k and a sink t_k. The flow of commodity k along a directed arc (x,y) is $y^k_{(x,y)}$, $(k = 1, \ldots, q)$ and these $y^k_{(x,y)}$, $(k = 1, \ldots, q)$ must satisfy the capacity constraints

$$\sum_{k=1}^{q} y^k_{(x,y)} \leq b_{(x,y)} \quad [(x,y) \in \mathcal{A}]$$

The Kirchhoff conservation requirements must be satisfied for each commodity $k(k = 1, \ldots, q)$ and for all $x \in N$ as

$$\sum_{n \in N} \left[y^k_{(x,n)} - y^k_{(n,x)} \right] = \begin{cases} v_k \text{ if } x = s_k \\ 0 \text{ if } x \neq s_k, t_k \\ -v_k \text{ if } x = t_k \end{cases}$$

where v_k is the total flow of commodity k from s_k to t_k.

The multicommodity, maximum-flow problem as formulated by Ford and Fulkerson[1] is as follows: Define the set $P^k = \left\{ P^{(k)}_j \mid P^{(k)}_j \text{ is a chain connecting } s_k \text{ and } t_k \right\}$. Now let P be the union of the sets $P^k (k = 1, \ldots, q)$. Further, let $P^{(1)}_1$, $P^{(1)}_2, \ldots, P^{(k)}_j, \ldots, P^{(q)}_n$ be the enumeration of the chains $P^{(k)}_j \in P$ such that the subscript j is sufficient to identify the chain, its origin-destination pair, and the commodity with which it is associated.

Thus the kth set is defined as

$$J_k = \left\{ j \mid P^{(k)}_j \text{ is a chain from } s_k \text{ to } t_k \right\}, k = 1, \ldots, q$$

The arc-chain incidence matrix is

$$A = [a_{ij}]$$

where

$$a_{ij} = \begin{cases} 1 \text{ if } \mathcal{A}_i \in P^{(k)}_j \\ 0 \text{ otherwise} \end{cases}$$

for $i = 1, \ldots, m$; $j = 1, \ldots, n$.

Consider the network used as an example in Ref 1, augmented by s_k and $t_k (k = 1, 2)$, with source s_1 and sink t_1 for commodity 1 and source s_2 and sink t_2 for commodity 2. Figure 1 illustrates the network and Fig. 2 shows the arc-chain incidence matrix A.

Letting $x^{(k)}_j$ $(j = 1, \ldots, n)$ be the flow of commodity k in chain $P^{(k)}_j$ $(j = 1, \ldots, n$; k implicit) and b_i the flow capacity of $\mathcal{A}_i$, the multicommodity, maximum-flow linear program is:

Maximize

$$\sum_{j=1}^{n} x^{(k)}_j$$

subject to capacity constraints

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i$$

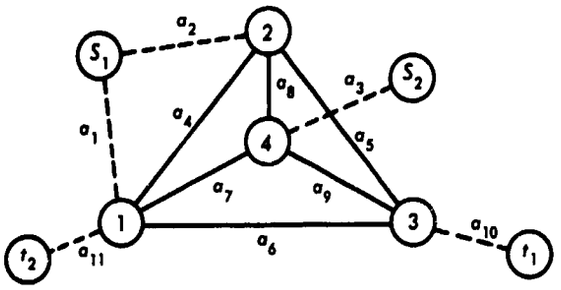


Fig. 1—Network A

	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	P_{12}	P_{13}	P_{14}	P_{15}
a_1	1	1	1	1	1										
a_2						1	1	1	1	1					
a_3											1	1	1	1	1
a_4			1		1			1		1		1			1
a_5			1	1		1								1	1
a_6	1							1	1				1	1	
a_7		1		1					1	1	1				
a_8				1	1		1		1			1		1	
a_9		1			1		1			1			1		1
a_{10}	1	1	1	1	1	1	1	1	1	1					
a_{11}											1	1	1	1	1
	Commodity 1										Commodity 2				

Fig. 2—Arc-Chain Incidence Matrix A

for $i = 1, \ldots, m$.

Thus the objective is to maximize flow over all possible chains from origins to their respective destinations subject to the capacity constraints of the arcs.

The number of variables in the aforementioned linear program is very large since the number of possible chains is very large in most applications. The procedure proposed by Ford and Fulkerson[1] is to treat the nonbasic variables implicitly; i.e., nonbasic chains are not enumerated. The column vector to enter the basis is generated by applying the simplex multipliers to the arcs as pseudo costs and selecting the candidate chain using the shortest chain algorithm.*

The remainder of this paper is concerned with an extension of this procedure to allow the addition of resource constraints on the flow over the network. The maximum-flow problem will be discussed first, followed by a minimum-cost formulation. A brief description of experience with a computer program using this procedure concludes the paper.

Extension To Include Resource Constraints

In the linear programming problem stated previously, flow is to be maximized subject to the constraints imposed by the capacities of the individual arcs of the network. In some applications additional constraints on flow are imposed by the limited availability of resources used jointly by two or more arcs of the network. An example of this type of network, which will be used throughout the remainder of this paper, is a transportation network.

It is clear that the simultaneous consideration of both types of constraints is an important problem in transportation networks. Roadways, rail lines, etc, have capacity limitations that may limit the maximum movement of men and materials, particularly in less-developed areas. The vehicles and resources

*Professor Mandell Bellmore of The Johns Hopkins University and Mr. Donald Boyer of the Logistics Research Project, The George Washington University, have developed computer programs to solve the problem using this procedure.

available to use the network can actually impose a greater constraint on total movement than the arc capacities. In a highly developed transportation system the capacity of the network may greatly exceed that required; the effective limitations of movement result from too few vehicles or other resources.

To effect this simultaneous consideration of resources as related to the network, the equipment required to sustain a flow of 1 unit across an arc (x, y) could be considered. For example, consider some of the equipment that might be necessary over the network:

(1) General-purpose trucks

(2) Tank trucks

(3) Trains (locomotives and cars)

(4) Barges

(5) Mechanics

(6) Drivers, laborers, etc

. . . .

(p) Other resources

Such a list may be as detailed or as aggregated as desired. For example, standard notional trains consisting of a locomotive and its complement of rail cars may be used, or the resources may be considered separately. Some notional truck combining all the various trucks available may be considered, or general-cargo trucks and tank trucks may be listed separately.

Let the resource matrix for commodity k be

$$R^k = [r_{is}^k] \quad (i = 1, \quad ; s = 1, \ldots, p)$$

where r_{is}^k is the quantity of resource s required to sustain a unit flow of commodity k over arc i; $r_{is}^k \geq 0$. Note that for some arc commodity combinations

$$r_{is}^k = \infty \quad (s = 1, \ldots, p)$$

e.g., if the arc represents a pipeline and the commodity is passengers.

Letting ρ_s be the quantity of resource s available (e.g., in inventory) for assignment to the network $(s = 1, \ldots, p)$, the linear programming problem with resource constraints is:

maximize

$$\sum_{j=1}^{\ddot{n}} x_j^{(k)}$$

subject to (a) capacity constraints

$$\sum_{j=1}^{n} a_{ij} x_j^{(k)} \leq b_i$$

for $i = 1, \ldots, m$, and (b) resource constraints

$$\sum_{i=1}^{m} \sum_{k=1}^{q} \sum_{j \in J_k} a_{ij} x_j^{(k)} r_{is}^k \leq \rho_s$$

for $s = 1, \ldots, p$.

In matrix form the flow-maximizing linear program with arc capacity and resource constraints is:
maximize

$$\sum_{j=1}^{n} x_j^{(k)}$$

subject to $\hat{A}X \leq b$ where $\hat{A}$ is a matrix ($m + p \times n$) formed of two submatrices A and E.

$$\hat{A} = \left[\frac{A}{E}\right]$$

$A = [a_{ij}]$ is the arc-chain incidence matrix. $E = [e_{sj}]$ is the resource-chain requirement matrix where

$$e_{sj} = \sum_{i=1}^{m} r_{is}^k a_{ij}$$

for k where $P_j^{(k)}$ connects s_k and t_k, $s = 1, \ldots, p$.

$$X = \text{col.} \left(x_1^{(k)}, x_2^{(k)}, \ldots, x_{m+p}^{(k)} \right)$$

and

$$b = \text{col.} \, (b_1, b_2, \ldots, b_m, \rho_1, \rho_2, \ldots, \rho_p)$$

In words, the $\hat{A}$ matrix is composed of column vectors that represent chains from s_k to t_k in a natural way. The chain is identified by the arcs that form it and by the resources required to sustain a unit flow from s_k to t_k. A typical column vector of $\hat{A}$ is

$$\hat{A}_j = \text{col.} \, (a_{1j}, a_{2j}, \ldots, a_{mj}, e_{1j}, e_{2j}, \ldots, e_{pj})$$

Solution Using the Chain Generation Technique

The flow-maximizing linear program with arc capacity and resource constraints will be quite large for most applications and will, in addition, require considerable preliminary computation to obtain the coefficients of the A matrix. (The authors have solved several small problems using a standard linear programming code.) It will now be shown that the column generation procedure suggested by Ford and Fulkerson[1] can be modified to apply to the problem extended to include resource constraints so that it is never necessary to form the A matrix explicitly.

The linear program described previously in matrix form is now modified to a standard linear programming form. (An attempt has been made throughout this discussion to follow the notation used by Hadley.[5]) The inequalities are changed to equalities and $m+p$ slack variables are added. A cost or profit vector is included in the calculation of the objective function. The problem then becomes:

maximize

$$CX = z$$

subject to

$$AX = b$$

where $A = [\hat{A} \mid I_{m+p}]$

$C = (c_1, \ldots, c_{n+m+p})$

c_j = the profit obtained for each unit of flow achieved on $P_j^{(k)}$ *

$X = \text{col.}\ (x_1^{(k)}, \ldots, x_{n+m+p}^{(k)})$

An initial basic solution is

$$B_0 X_b = b$$

with objective function

$$C_b X_b = z = 0$$

*In the previous formulation of this problem the value of the c_j's ($j = 1, \ldots, n$) was assumed to be 1. That is, the value of a unit of flow achieved on any chain is equal to that achieved on any other chain. This is the usual case. If any $c_j \neq 1$ then one must require that $c_r = c_s$ for all $r,\ s \in J_k$, for $k = 1, \ldots, q$, in order for the proposed algorithm to be valid. For $c_r \neq c_s$ ($r,\ s \in J_k$) implies that of the chains from s_k to t_k some are preferred over others.

where $B_0 = I_{m+p}$

$C_b = (c_{j1}, c_{j2}, \ldots, c_{jm+p})$

$X_b = \text{col.}\ (x_{j1}^{(k)}, x_{j2}^{(k)}, \ldots, x_{jm+p}^{(k)})$

such that $j_1, j_2, \ldots, j_{m+p}$ are selected so that C_b and X_b conform properly to the B_0 matrix, which is the basis.

Thus one begins with a feasible solution, and the simplex procedure is fundamentally a method for improving this solution by selection vectors from the A matrix to enter the basis. Given a feasible solution to the linear programming problem as expressed previously, the value of the new objective function (following the introduction of a new vector) is:[5]

$$\hat{z} = z + x_{Br}/y_{rj}\,(c_j - z_j) = z + \theta\,(c_j - z_j)$$

Thus, for a suitably selected θ the value of the new objective function will be greater than the old if, and only if,

$$c_j - z_j > 0 \qquad (1)$$

Now, since B is a basis, every A_j (a candidate to enter the basis) can be expressed as

$$A_j = y_1 B_1 + y_2 B_2 + \ldots + y_{m+p} b_{m+p} = \sum_{i=1}^{m+p} y_i B_i$$

the B_i's being the $m+p$ column vectors of B and the y_i's the unique scalars such that A_j is a linear combination of the B_i. Now let

$$Y_j = \begin{bmatrix} y_{1j} \\ y_{2j} \\ \cdot \\ \cdot \\ \cdot \\ y_{m+p,j} \end{bmatrix}$$

so that

$$A_j = B Y_j$$

and

$$Y_j = B^{-1} A_j \qquad (2)$$

Now the present objective function (i.e., before the introduction of a new A_j) is in terms of the present basis:

$$z = c_B x_B = \sum_{i=1}^{m+p} c_{Bi} x_{Bi}^{(k)}$$

But since A_j is expressible in terms of this basis, there is for each A_j, a candidate to enter the basis, a new $\hat{z}_j$ as follows:

$$\hat{z}_j = y_{1j} c_{B1} + y_{2j} c_{B2} + \cdots y_{m+p,j} c_{B_{m+p}}$$

or

$$\hat{z}_j = c_B Y_j \tag{3}$$

Thus, substituting Eq 3 in Eq 1,

$$c_j - c_B y_j > 0 \tag{4}$$

and substituting Eq 2 in Eq 4,

$$c_j - c_B B^{-1} A_j > 0$$

Now, bringing c_j to the right-hand side and multiplying through by -1,

$$c_B B^{-1} A_j < c_j \tag{5}$$

which is the simplex rule and a criterion for choosing a new A_j to enter the basis. If $c_B B^{-1} A_j < c_j$ then A_j can be brought into the basis and the new value of the objective function will be equal to or greater than the preceding value.

Applying the simplex criterion in the stated maximization problem, one may choose

$$\min_j (z_j - c_j)$$

or

$$\min_j (c_B B^{-1} A_j - c_j)$$

and when all

$$c_B B^{-1} A_j - c_j \geq 0$$

an optimum solution has been found.

It is the hypothesis of the Ford and Fulkerson suggested solution to the multicommodity problem (and of the present resource-constrained algorithm) that the shortest chain algorithm can be used to develop the A_j that will satisfy the simplex rule. Further, if the shortest chain algorithm can find no chain satisfying the requirement, an optimum has been reached.

Let $c_B B^{-1} = (\alpha_1, \alpha_2, \ldots, \alpha_m, \pi_1, \pi_2, \ldots, \pi_p)$. The shortest chain in the multicommodity problem without resource constraints may be found by assigning the α_i to the corresponding arc i so that $d_{(x,y)} = \alpha_i$,* where $\mathcal{C}_i = (x,y)$; the shortest chain algorithm can then be used to find the chain connecting s_k to t_k $(k = 1, \ldots, q)$ such that $c_B B^{-1} A_j - c_j$ is minimized. Further, if the shortest chain $c_B B^{-1} A_j \geq c_j$ then no chain exists for which $c_B B^{-1} A_j < c_j$ and the optimum has been reached.

Now in the resource-constrained problem this procedure must be modified since all the elements of $c_B B^{-1}$ cannot now be assigned directly to arcs. This is because π_1 to π_p correspond to resources, not arcs. Therefore a "pseudo cost" must be devised that can be assigned directly to arcs such that the shortest chain will be the chain with minimum $(c_B B^{-1} A_j - c_j)$. Now

$$c_B B^{-1} A_j = (\alpha_1, \alpha_2, \ldots, \alpha_m, \pi_1, \pi_2, \ldots, \pi_p) \begin{bmatrix} a_{1j} \\ a_{2j} \\ \vdots \\ a_{mj} \\ e_{1j} \\ e_{2j} \\ \vdots \\ e_{pj} \end{bmatrix}$$

where

$$A_j = \begin{bmatrix} a_{1j} \\ a_{2j} \\ \vdots \\ a_{mj} \\ e_{1j} \\ e_{2j} \\ \vdots \\ e_{pj} \end{bmatrix} = \begin{bmatrix} a_{1j} \\ a_{2j} \\ \vdots \\ a_{mj} \\ \sum_{i=1}^{m} r_{i1}^k a_{ij} \\ \vdots \\ \sum_{i=1}^{m} r_{ip}^k a_{ij} \end{bmatrix}$$

*If, however, any $\alpha_i < 0$ or $\pi_i < 0$ the shortest chain algorithm may fail. The slack vector corresponding to the negative α_i or π_i should therefore be brought in. This is a unit vector (U_i) with a 1 in the ith position (a 1 in the $m+i$th position in the case of $\pi_i < 0$) and zeros elsewhere. For if $\alpha_i < 0$, then $c_B B^{-1} U_i < c_i$, since $c_j \geq 0$ $(j = 1, \ldots, n+m+p)$.

for k where $p_j^{(k)}$ connects s_k $\quad$ t_k. Thus

$$c_B B^{-1} A_j = \sum_{i=1}^{m} \left[\alpha_i a_{ij} + \sum_{s=1}^{p} \pi_s r_{is}^k a_{ij} \right]$$

and the contribution of each arc i, member of the chain $P_j^{(k)}$, is

$$\alpha_i a_{ij} + \sum_{s=1}^{p} \pi_s r_{is}^k a_{ij} \text{ (for } k, \text{ where } P_j^{(k)} \text{ connects } s_k \text{ and } t_k).$$

Define a real numbered function d_i^k as

$$d_i^k = \alpha_i a_{ij} + \sum_{s=1}^{p} \pi_s r_{is}^k a_{ij}$$

and assign d_i^k to each arc i. The shortest chain algorithm[6] may then be used to find

$$\min_j \left[\sum_{Q_i \in P_j} d_i^k \right] = \min_j \left[\sum_{Q_i \in P_j} \left(\alpha_i a_{ij} + \sum_{s=1}^{p} \pi_s r_{is}^k a_{ij} \right) \right]$$

$$= \min_j c_B B^{-1} A_j \text{ (for } k = 1, \ldots, q)$$

Thus the candidate A_j is found to enter the basis. The column vector to leave the basis may be determined in the normal simplex manner. The procedure for determining the maximum multicommodity network flows with resource constraints is as follows:

(1) Form an initial basis $B_0 = I_{m+p}$.

(2) Calculate B^{-1}, $c_B B^{-1} = (\alpha_1, \ldots, \alpha_m, \pi_1, \ldots, \pi_p)$, and $z = c_B B^{-1} b$.

(3) Evaluate $(\alpha_1, \ldots, \alpha_m, \pi_1, \ldots, \pi_p)$ as follows:

(a) If any $\alpha_i < 0$, enter into the basis the slack vector for that arc, which is a unit vector with a 1 in the ith position, zeros elsewhere, and go to step 2.

(b) If any $\pi_i < 0$, enter into the basis the slack vector for that resource, which is a unit vector with a 1 in the $(m + i)$th position, zeros elsewhere, and go to step 2.

(c) If no $\alpha_i, \pi_i < 0$, go to step 4.

(4) Determine the $\min_j [c_B B^{-1} A_j - c_j]$ as follows:

(a) For $k = 1, \ldots, q$, calculate and assign d_i^k to arcs $i, \ldots, m$ and find the shortest chain from s_k to t_k.

(b) From the q shortest chains, select that chain with the minimum $[c_B B^{-1} A_j - c_j]$.*

(5) If the minimum chain results in $c_B B^{-1} A_j \geq c_j$, terminate the algorithm since there exists no chain that will increase the value of z; otherwise go to step 6.

(6) Enter the minimum A_j of step 5 into the basis. Determine the vector to leave the basis in the normal simplex manner. Go to step 2.

3. THE COST-MINIMIZATION PROBLEM

In many applications the objective is to minimize the cost of resource utilization to meet fixed delivery requirements at given destinations. Tomlin has proposed such an extension to the arc-chain formulation of the maximum-flow linear program.[2] In this section of the paper Tomlin's formulation is modified to include resource constraints and a revised column generation scheme as a solution procedure is proposed.

The minimum-cost, multicommodity flow problem with resource constraints may be formulated as follows:†

minimize

$$\sum_{j=1}^{n} c_j x_j^{(k)}$$

subject to (a) capacity constraints

$$\sum_{j=1}^{n} a_{ij} x_j^{(k)} \leq b_i \quad \text{for } i = 1, \ldots, m$$

(b) resource constraints

$$\sum_{i=1}^{n_i} \sum_{k=1}^{q} \sum_{j \in J_k} a_{ij} x_j^{(r)} r_{is}^k \leq \rho_s \quad \text{for } s = 1, \ldots, p$$

and (c) delivery requirements

$$\sum_{j \in J_k} x_j^{(k)} = \lambda_k \quad \text{for } k = 1, \ldots, q$$

where λ_k is the delivery requirement at t_k $(k = 1, \ldots, q)$, $(\lambda_k \geq 0)$.

*The algorithm may be modified to accept the first chain where $c_B B^{-1} A_j < c_j$.

†The notation used in this section parallels, where applicable, that used in the preceding maximum-flow section.

The cost coefficient c_j may be defined as:

$$c_j = \sum_{i=1}^{m} r_i a_{ij} + \sum_{s=1}^{p} \phi_s e_{sj} \quad \text{for } j = 1, \ldots, n$$

$$= \sum_{i=1}^{m} r_i a_{ij} + \sum_{s=1}^{p} \sum_{i=1}^{m} \phi_s r_{is}^{k} a_{ij} \quad (\text{for } j = 1, \ldots, n;\ k \text{ where } P_j^{(k)} \text{ connects } s_k \text{ and } t_k)$$

where r_i is the cost (or toll) for a unit flow over arc i, and ϕ_s is the cost of using a unit of resource s.

Define the matrices G and $\hat{\hat{A}}$ as follows: G is a commodity delivery incidence matrix $(q \times n)$

$$G = [g_{kj}]$$

where

$$g_{kj} = \begin{cases} 1 \text{ if } j \epsilon J_k^{*} \\ 0 \text{ otherwise} \end{cases}$$

$\hat{\hat{A}}$ is a matrix $(m + p + q \times n)$ formed of the submatrices A, E, and G as follows:

$$\hat{\hat{A}} = \begin{bmatrix} A \\ \hline E \\ \hline G \end{bmatrix}$$

Now the typical column of $\hat{\hat{A}}$ is

$$\hat{\hat{A}}_j = \text{col. } (a_{1j}, \ldots, a_{mj}, e_{1j}, \ldots, e_{pj}, g_{1j}, \ldots, g_{qj})$$

In the matrix notation the cost-minimization problem is then to minimize

$$CX$$

subject to

$$AX = b$$

where

$$A = \left[\hat{\hat{A}} \mid I_{m+p+q} \right]$$

and

$$C = (c_1, \ldots, c_{n+m+p+q})$$

The c_j are the costs associated with the chains $(j = 1, \ldots, n)$, the slack variables $(j = n+1, \ldots, n+m+p)$, and the artificial variables $(j = n+m+p+1, \ldots, n+m+p+q)$.

*Note that all columns of g are unit vectors. No chain meeting the Kirchhoff conservation requirements may be a member of more than one of the sets J_k, $k = 1, \ldots, q$.

Adopting the standard two-phased solution procedure for the minimum-cost formulation, the Phase I procedure minimizes to zero the value of

$$\sum_{j=n+m+1}^{j=n+m+p+q} x_j^{(k)}$$

to obtain an initial basic feasible solution. This effectively assigns a cost of 1 to the artificial variables and a cost of zero to the other variables in Phase I. Phase II begins with the basic feasible solution determined in Phase I and proceeds to minimize

$$\sum_{j=1}^{j=n+m+p+q} c_j x_j^{(k)}$$

Phase I

In Phase I I_{m+p+q} may be used as the initial basis and the simplex rule is to enter a chain in the basis if, and only if,

$$c_j - c_B B^{-1} A_j < 0$$

where

$$c_B B^{-1} = (\alpha_1, \ldots, \alpha_m, \pi_1, \ldots, \pi_p, \sigma_1, \ldots, \sigma_q)$$

so that the simplex multipliers α_i are associated with the arcs, the π_s are associated with the resources, and the σ_k are associated with the artificial variables. Thus the vector A_j is entered if

$$c_j - c_B B^{-1} A_j = - \sum_{i=1}^{m} \alpha_i a_{ij} - \sum_{s=1}^{p} \pi_s e_{sj} - \sum_{k=1}^{q} \sigma_k g_{kj} < 0$$

$$= - \sum_{i=1}^{m} \alpha_i a_{ij} - \sum_{s=1}^{p} \sum_{i=1}^{m} \pi_s a_{ij} r_{is}^k - \sum_{k=1}^{q} \sigma_k g_{kj} < 0$$

Since each chain is associated with one and only one of the $t_k (k = 1, \ldots, q)$, and

$$\sum_{k=1}^{q} \sigma_k g_{kj} = \sigma_k$$

then A_j is to enter the basis if

$$- \sum_{i=1}^{m} a_{ij} \left[\alpha_i + \sum_{s=1}^{p} \pi_s r_{is}^k \right] < \sigma_k$$

Thus the contribution of each arc to $c_j - c_B B^{-1} A_j$ in Phase I is

$$d_i^k = -\left[\alpha_i + \sum_{s=1}^{p} \pi_s r_{is}^k\right]$$

One may then use the shortest chain algorithm as in the maximum-flow case to find, for each k,

$$\min_j \left[\sum_{a_i \epsilon P_j} d_i^k\right] = \min_j \left[\sum_{a_i \epsilon P_j} \left(-\alpha_i - \sum_{s=1}^{p} \pi_s r_{is}^k\right)\right]$$

$$= \min_j c_B B^{-1} A_j \text{ over all } k$$

Thus, one may find the candidate A_j to enter the basis. The column vector to leave the basis may be determined in the usual manner.

Should any α_i or π_i be positive one can, in a manner analogous to the maximum-flow case, bring in the corresponding slack variable. Should the minimum $c_B B^{-1} A_j$ be greater than or equal to zero, there is no feasible solution and the procedure would be terminated. If there is a feasible solution, Phase I will terminate when all artificial variables have vanished and $z = 0$.

Phase II

Phase II begins with a basic feasible solution with all artificial variables equal to zero. Its simplex criterion states that a chain, represented by the vector A_j, may enter the basis if, and only if,

$$c_j - c_B B^{-1} A_j < 0$$

In Phase II,

$$c_j - c_B B^{-1} A_j = \sum_{i=1}^{m} r_i a_{ij} + \sum_{s=1}^{p} \phi_s e_{sj} - \sum_{i=1}^{m} \alpha_i a_{ij} - \sum_{s=1}^{p} \pi_s e_{sj} - \sum_{k=1}^{q} \sigma_k g_{kj}$$

$$= \sum_{i=1}^{m} a_{ij} (r_i - \alpha_i) + \sum_{s=1}^{p} e_{sj} (\phi_s - \pi_s) - \sum_{k=1}^{q} \sigma_k g_{kj}$$

$$= \sum_{i=1}^{m} a_{ij} (r_i - \alpha_i) + \sum_{s=1}^{p} \sum_{i=1}^{m} a_{ij} r_{is}^k (\phi_s - \pi_s) - \sum_{k=1}^{q} \sigma_k g_{kj}$$

$$= \sum_{i=1}^{m} a_{ij} \left[(r_i - \alpha_i) + \sum_{s=1}^{p} r_{is}^k (\phi_s - \pi_s)\right] - \sum_{k=1}^{q} \sigma_k g_{kj}$$

Thus

$$d_i^{k'} = r_i - \alpha_i + \sum_{s=1}^{p} r_{is}^k (\phi_s - \pi_s)$$

is the analogue of d_i^k in the maximum-flow case and may be assigned as a pseudo cost to each arc. The shortest chain, i.e., the chain with the least,

$$\sum_{i=1}^{m} d_i^{k'} - \sigma_k < 0$$

for $k = 1, \ldots, q$, may then be entered in the basis.

The Phase II simplex rule is, then, enter the vector A_j if

$$\mathfrak{C}_{i \in P_j} \sum d_i^{k'} < \sigma_k$$

or

$$\sum_{i=1}^{m} a_{ij} \left[(r_i - \alpha_i) + \sum_{s=1}^{p} r_{is}^k (\phi_s - \pi_s) \right] < \sigma_k$$

As in Phase I, a positive α_i or π_s will indicate that the corresponding slack variable should enter the basis. Phase II is terminated and the value of z is minimized when, for the minimum j,

$$\mathfrak{C}_{i \in P_j} \sum d_i^{k'} \geq \sigma_k$$

4. COMPUTATIONAL EXPERIENCE

A computer program for the IBM 7044 has been developed to solve both the maximum-flow and minimum-cost problems. This test program is an all-in-core routine that updates the basis inverse each iteration. A series of test cases has been solved successfully. The largest of the series had 108 arcs, 9 types of resources, and 3 commodities. Several variations of this case have been tested including both network-constrained and resources-constrained problems. Two typical problems were also solved by enumerating the possible chains for the LP 3 linear programming code. A comparison of the results of the two methods indicates that the proposed algorithm is accurate and significantly faster than the standard code.

REFERENCES

CITED REFERENCES

1. L. R. Ford Jr. and D. R. Fulkerson, "A Suggested Computation for Maximal Multi-Commodity Network Flows," Mgt. Sci., Oct 58.
2. J. A. Tomlin, "Minimum-Cost Multi-Commodity Network Flows," Opns. Res., Dec 66.
3. William S. Jewell, "A Primal-Dual Multi-Commodity Flow Algorithm," ORC 66-24, Operations Research Center, University of California, Berkeley, Sep 66.
4. G. B. Dantzig and P. Wolfe, "The Decomposition Algorithm for Linear Programming," Econometrica, 29: 767–78 (1961).
5. G. Hadley, Linear Programming, Addison-Wesley Publishing Co., Inc., Reading, Mass., 1962.
6. L. R. Ford Jr. and D. R. Fulkerson, Flows in Networks, Princeton University Press, Princeton, N.J., 1962.

ADDITIONAL REFERENCES

Boyer, Donald D., "A Modified Simplex Algorithm for Solving the Multi-Commodity Maximum Flow Problem," TM-14930, The George Washington University Logistics Research Project, Washington, D.C., Mar 68.

Busacker, R. G., et al, "Three General Network Flow Problems and Their Solutions," RAC-SP-183, Research Analysis Corporation, Nov 62.

Fitzpatrick, G. R., et al, "Programming the Procurement of Airlift and Sealift Forces: A Linear Programming Model for Analysis of the Least Cost Mix of Strategic Deployment Systems," Naval Research Logistics Quart., 14 (2): (1967).

Rao, M. R., and S. Zionts, "Allocation of Transportation Units to Alternative Trips—A Column Generation Scheme with Out-of-Kilter Subproblems," Opns. Res., Jan–Feb 68.

Sakarovitch, M., "The Multi-Commodity Maximum Flow Problem," Operations Research Center, University of California, Berkeley, Dec 66.

REPRINT OF

RAC PUBLICATION RAC-P-52, APRIL 1969

AN EXTENSION TO THE MULTICOMMODITY NETWORK FLOW MODEL TO PERMIT SUBSTITUTION OF RESOURCES

LOGISTICS DEPARTMENT
RAC-P-52
Published April 1969

An Extension to the Multicommodity Network Flow Model To Permit Substitution of Resources

by
John E. Cremeans
Robert A. Smith
Gene R. Tyndall

RAC Research Analysis Corporation
McLean, Virginia

Preceding page blank

CONTENTS

An Extension to the Multicommodity Network Flow Model To Permit Substitution of Resources

Preceding page blank

ABSTRACT

In a prior paper (RAC-P-44, "A Model for Optimal Multicommodity Network Flows with Resource Allocation,"[1]) a methodology for determining optimal multicommodity flows over a capacitated network with simultaneous consideration of network and resource constraints was developed. This paper describes an extension to this basic work to incorporate the ability to handle resource substitutions. Substitution of resources is the capability to employ alternative resource combinations to accomplish flows in the network. The extended model permits the selection of an optimal set of resources and chains to accomplish required flows in a network.

INTRODUCTION

In an earlier paper (RAC-P-44[1]) a model was presented for the solution of multicommodity network flow optimization problems with resource constraints. Included in the paper was a description of computational experience to date. The model formulation was based on an extension to the shortest-chain procedure for solving multicommodity flow problems suggested originally by Ford and Fulkerson[2] in 1958 and further developed by Tomlin,[3] Bellmore,[4] Boyer,[5] and others.

The extended formulation considers the problem in which resources must be assigned to the arcs of a chain in order to sustain a flow over that chain. For example, consider a transportation-network problem in which trucks, drivers, gasoline, trains, barges, etc, must be allocated to the various routes if any movement is to be accomplished. The extended formulation thus includes (in addition to arc-capacity constraints) resource-utilization constraints that prevent the assignment of more of any type of resource than is available. Therefore, in the maximum-flow mode, routes are selected and resources allocated to attain maximum network flow. In the minimum-cost mode, routes are selected and resources allocated so as to minimize the cost of meeting flow requirements subject to the arc capacity and resource-utilization constraints.

This paper presents a modification of the initial extended formulation to allow for the substitution of resources. In the previous paper only one combination of resources was permitted to be applied to an arc in order to move one unit of commodity k over arc i. In economic terms the arc-commodity pair is similar to a production function with constant returns to scale and fixed technical coefficients (see R. G. D. Allen, Macro-Economic Theory,[6] p 36). In some applications this may be a significant limitation. Consider again a transportation network. A highway arc might be considered for the transport of manufactured products. Closed vans with a driver and an alternate driver

might be the most efficient combination of resources. A combination of a van and one driver would be less efficient, perhaps, since more rest periods would be required, but it is nevertheless a feasible combination. Similarly a third alternative would be the utilization of stake and platform trucks with containers, possibly more expensive than the first two alternatives but still feasible.

Specific inventories of trucks and drivers may exist and the objective might be to assign these sets of resources in the most efficient way over all arcs even if some arcs or commodities are assigned a less than most efficient set of resources.

An extension to the multicommodity network flow model with resource constraints is proposed that permits alternative resource combinations to be employed to accomplish flows over arcs and chains.

NOTATION

To familiarize the reader with the notation used in the development of the extended multicommodity network flow model, this section of the report is devoted to a brief review of the mathematical structure of the resource-constrained multicommodity network flow model. This section, as well as the following section on substitution of resources, emphasizes the minimum-cost mode formulation rather than the maximum-flow mode (minimum cost being the more detailed of the two similar formulations).

Consider the multimode multicommodity network $G(N, \mathcal{A})$. N is the set of all the nodes of the network. $\mathcal{A}$ is the subset of all ordered pairs (x, y) of the elements of N that are arcs of the network. $\mathcal{A}_1, \ldots, \mathcal{A}_m$ is an enumeration of the arcs. Each arc has an associated capacity $b_{(x,y)} \geq 0$ and an associated cost (or distance) $d_{(x,y)} \geq 0$.

For each commodity k $(k = 1, \ldots, q)$ there is a source s_k and a sink t_k. The flow of commodity k along a directed arc (x,y) is $y^k_{(x,y)}$, $(k = 1, \ldots, q)$, and these $y^k_{(x,y)}$, $(k = 1, \ldots, q)$ must satisfy the capacity constraints

$$\sum_{k=1}^{q} y^k_{(x,y)} \leq b_{(x,y)} \quad [(x,y) \in \mathcal{A}]$$

Define the set $P^k = \{ P_j^{(k)} \mid P_j^{(k)}$ is a chain connecting s_k and $t_k \}$. Now let P be the union of the sets P^k $(k = 1, \ldots, q)$. Further, let $P_1^{(1)}, P_2^{(1)}, \ldots, P_j^{(k)}$,

$\ldots, P_n^{(q)}$ be the enumeration of the chains $P_j^{(k)} \in P$ such that the subscript j is sufficient to identify the chain, its origin-destination pair, and the commodity with which it is associated.

Thus the kth set is defined as

$$J_k = \left\{ j \mid P_j^{(k)} \text{ is a chain from } s_k \text{ to } t_k \right\}, \quad k = 1, \ldots, q$$

The arc-chain incidence matrix is

$$\hat{A} = [a_{ij}]$$

where

$$a_{ij} = \begin{cases} 1 \text{ if } \mathcal{A}_i \in P_j^{(k)} \\ 0 \text{ otherwise} \end{cases}$$

for $i = 1, \ldots, m$; $j = 1, \ldots, n$.

Let the resource matrix for commodity k be

$$R^k = [r_{is}^k] \quad (i = 1, \ldots, m; \; s = 1, \ldots, p)$$

where r_{is}^k is the quantity of resource s required to sustain a unit flow of commodity k over arc i; $r_{is}^k \geq 0$. Note that for some arc-commodity combinations

$$r_{is}^k = \infty \quad (s = 1, \ldots, p)$$

e.g., if the arc represents a pipeline and the commodity is passengers.

Let $x^{(k)}$ $(j = 1, \ldots, n)$ be the flow of commodity k in chain $P_j^{(k)}$ $(j = 1, \ldots, n;$ k implicit), b_i the flow capacity of $\mathcal{A}_i$, and ρ_s the quantity of resource s available (e.g., in inventory) for assignment to the network $(s = 1, \ldots, p)$. The minimum-cost multicommodity flow problem with resource constraints may be formulated as follows:

minimize

$$\sum_{j=1}^{n} c_j x_j^{(k)}$$

subject to (a) capacity constraints

$$\sum_{j=1}^{n} a_{ij} x_j^{(k)} \leq b_i \text{ for } i = 1, \ldots, m$$

(b) resource constraints

$$\sum_{i=1}^{m} \sum_{k=1}^{q} \sum_{j \in J_k} a_{ij} x_j^{(k)} r_{is}^k \leq \rho_s \text{ for } s = 1, \ldots, p$$

and (c) delivery requirements

$$\sum_{j \in J_k} x_j^{(k)} = \lambda_k \text{ for } k = 1, \ldots, q$$

where λ_k is the delivery requirement at t_k $(k = 1, \ldots, q)$, $(\lambda_k \geq 0)$.

The cost coefficient c_j may be defined as

$$c_j = \sum_{i=1}^{n} r_i a_{ij} + \sum_{s=1}^{p} \phi_s e_{sj} \text{ for } j = 1, \ldots, n$$

$$= \sum_{i=1}^{m} r_i a_{ij} + \sum_{s=1}^{p} \sum_{i=1}^{m} \phi_s r_{is}^{k} a_{ij} \text{ (for } j = 1, \ldots, n; \; k \text{ where } P_j^{(k)} \text{ connects } s_k \text{ and } t_k)$$

where r_i is the cost (or toll) for a unit flow over arc i, and ϕ_s is the cost of using a unit of resource s

Define the matrices G and $\hat{\hat{A}}$ as follows: G is a commodity-delivery incidence matrix $(q \times n)$

$$G = [\mathcal{G}_{ij}]$$

where

$$\mathcal{G}_{kj} = \begin{cases} 1 \text{ if } j \in J_k \\ 0 \text{ otherwise} \end{cases}$$

$\hat{\hat{A}}$ is a matrix $(m + p + q \times n)$ formed of the submatrices A, E, and G as follows:

$$\hat{\hat{A}} = \begin{bmatrix} \hat{A} \\ \hline E \\ \hline G \end{bmatrix}$$

Now the typical column of $\hat{\hat{A}}$ is

$$\hat{\hat{A}}_j = \text{col.}\,(a_{1j}, \ldots, a_{mj}; \; e_{1j}, \ldots, e_{pj}; \; \mathcal{G}_{1j}, \ldots, \mathcal{G}_{qj})$$

In the matrix notation the cost-minimization problem is then to minimize

$$CX$$

subject to

$$AX = b$$

where

$$\left[A = \hat{\hat{A}} \,\middle|\, I_{m+p+q}\right]$$

and

$$C = (c_1, \ldots, c_{n+m+p+q})$$

The c_j's are the costs associated with the chains ($j = 1, \ldots, n$), the slack variables ($j = n + 1, \ldots, n + m + p$, and the artificial variables ($j = n + m + p + 1, \ldots, n + m + p + q$).

Adopting the standard two-phase solution procedure for the minimum-cost formulation, the Phase I procedure minimizes to zero the value of

$$\sum_{j=n+m+p+1}^{j=n+m+p+q} x_j^{(k)}$$

to obtain an initial basic feasible solution. This effectively assigns a cost of 1 to the artificial variables and a cost of zero to the other variables in Phase I. Phase II begins with the basic feasible solution determined in Phase I and proceeds to minimize

$$\sum_{j=1}^{j=n+m+p+q} c_j x_j^{(k)}$$

DISCUSSION

Substitution of Resources

In the previous formulation, for each arc-commodity pair a single combination of resources is required and represented by the vector, $R_i^k = (r_{i1}^k, r_{i2}^k, \ldots, r_{ip}^k)$, of the matrix R^k.

Define a new resource matrix: $T = [t_{ik}]$ ($i = 1, \ldots, m$; $k = 1, \ldots, q$) where $t_{ik} = \{\hat{R}_i^k \mid \hat{R}_i^k$ is any feasible resource vector for arc i, commodity $k\}$; $\hat{R}_i^k = (\hat{r}_{i1}^k, \ldots, \hat{r}_{ip}^k)$.

In words, each element of T is the set of alternative resource vectors for a movement of one unit of commodity k over arc i.

The contribution of each arc to $c_j - c_B B^{-1} A_j$ in Phase II of the minimum-cost procedure is

$$d_i^{k'} = r_i - \alpha_i + \sum_{s=1}^{p} r_{is}^k (\phi_s - \pi_s)$$

The possibility of employing alternative methods, i.e., alternative combinations of resources, affects this by allowing for a number of vectors R_i^k. Thus to find the minimum $c_j - c_B B^{-1} A_j$ one must find the minimum

$$\left[\sum_{i \in P_j} d_i^{k'}\right]$$

over the permissible $\hat{R}^k$ as well as over all feasible combinations of arcs. The elements ϕ_s $(s = 1, \ldots, p)$ are fixed for any problem and the elements π_s $(s = 1, \ldots, p)$ are fixed for any iteration. One may therefore find the vector

$$\bar{R}_i^k = \hat{R}_i^k \in t_{ik} \left[\sum_{s=1}^{p} \hat{r}_{is}^k (\phi_s - \pi_s) \right] \text{ for } i = 1, \ldots, m;\ k = 1, \ldots, q$$

The kth matrix of these minima may then be defined as:

$$\bar{R}^k = [\bar{r}_{is}^k] \ (i = 1, \ldots, m;\ s = 1, \ldots, p)$$

Each column vector $\bar{R}_i^k = (\bar{r}_{i1}^k, \ldots, \bar{r}_{ip}^k)$ is the alternative combination of resources such that

$$\sum_{s=1}^{p} \bar{r}_{is}^k (\phi_s - \pi_s)$$

is minimized for arc i and commodity k. Now $\bar{R}^k$ may be substituted in the minimum-cost procedure previously discussed and the appropriate A_i selected for entry into the basis. Thus new $\bar{R}^k$ is constructed for each commodity each iteration.

Summary of the Procedure

To summarize, the proposed procedure is:

(1) Calculate $C_B B^{-1} = (\alpha_1, \ldots, \alpha_m, \pi_1, \ldots, \pi_s, \sigma_1, \ldots, \sigma_q)$

(2) For each arc-commodity pair find the least-cost applicable resource vector, "cost" meaning cost in terms of the simplex multipliers and resource prices.

$$\bar{R}_i^k = \hat{R}_i^k \in t_{ik} \left[\sum_{s=1}^{p} \hat{r}_{is}^k (\phi_s - \pi_s) \right]$$

(3) For commodity $k = 1, \ldots, q$ calculate

$$d_i^{k'} = r_i - \alpha + \sum_{s=1}^{p} \bar{r}_{is}^k (\phi_s - \pi_s) \text{ for } i = 1, \ldots, m$$

and assign the d_i^k to the arc i as a pseudo cost.

(4) Using the shortest-chain algorithm find the chain with least

$$d_j^{k'} = \sum_{i=1}^{m} a_{ij} \left[r_i - \alpha_i + \sum_{s=1}^{p} \bar{r}_{is}^k (\phi_s - \pi_s) \right] \text{ for } k = 1, \ldots, q$$

(5) Find

$$\min_k \left[d_j^{k'} - \sigma_k \right]$$

(6) If the minimum $[d_j^k - \sigma_k] < 0$, the vector A_j is entered in the basis. If $[d_j^k - \sigma_k] \geq 0$, there is no chain that may improve the value of the objective function, and the procedure is terminated.

Validity of the Procedure

Consider the linear programming formulation of the substitution problem. It is identical to the original cost-minimization problem except that every column vector in the original problem will be replaced by

$$\alpha_{i \in P_j}^{\prod N(M_i)} \quad \begin{bmatrix} \text{where } N(M_i) \text{ is the number of alternate resource} \\ \text{vectors applying to arc } i \end{bmatrix}$$

alternate chains. The expanded substitution matrix will be many times larger than the original matrix, should either actually be enumerated.

It is claimed that the procedure outlined here will find the least-cost (in the sense previously described) vector to enter the basis. It should be noted that if the procedure does not find the least-cost vector, but some other vector, say the nth least-cost vector, the algorithm will progress toward an optimum solution in the early stages but will terminate early. That is, any vector that satisfies the simplex rule may be brought into the basis, but since the algorithm is terminated when the "shortest" chain does not satisfy the simplex rule, the validity of the procedure depends on the validity of the shortest-chain procedure.

Suppose that the chain produced as a candidate is not the shortest chain and there is some other candidate chain j^* for which

$$d_{j^*}^{k'} - \sigma_{k^*} < d_j^{k'} - \sigma_k$$

Two possibilities for this other chain exist:

(1) The shorter chain consists of the same arcs as our candidate chain but has different (allowable) resource vectors associated with one or more of these arcs.

(2) The shorter chain consists of different arcs altogether with some allowable set of resource vectors assigned to their respective arcs.

The first case is a chain that

$$d_{j*}^{k'} < d_j^{k'}$$

or that

$$\sum_{i=1}^{m} a_{ij}(r_i - \alpha_i) + \sum_{i=1}^{m} a_{ij} \sum_{s=1}^{p} \bar{r}_{is}^{k}(\phi_s - \pi_s) - \sigma_k$$

is less than

$$\sum_{i=1}^{m} a_{ij}(r_i - \alpha_i) + \sum_{i=1}^{m} a_{ij} \sum_{s=1}^{p} \bar{r}_{is}^{k}(\phi_s - \pi_s) - \sigma_k$$

but, since the first and last terms of each expression are identical, that is a claim that for at least one arc, common to both chains,

$$\sum_{s=1}^{p} \bar{r}_{is}^{k}(\phi_s - \pi_s) < \sum_{s=1}^{p} \bar{r}_{is}^{k}(\phi_s - \pi_s)$$

but since $\sum_{s=1}^{p} \bar{r}_{is}^{k}(\phi_s - \pi_s)$ $(i = 1, \ldots, m;\ k = 1, \ldots, q)$ is the minimum available (step 2), the claim that $d_{j*}^{k'} - \sigma_{k*} < d_j^{k'} - \sigma_k$ is inconsistent, and hence case 1 cannot occur. A true shortest chain must employ the least-cost allowable resources on each arc that is a member of the chain.

Case 2 resolves itself to a claim that there is some chain that uses the least-cost allowable resources on each of its member arcs and has a lower $(d_{j*}^{k'} - \sigma_{k*})$ than that of the candidate chain. Since the proposed procedure evaluates the pseudo cost of each arc incorporating the minimum resource costs [i.e., $\sum_{s=1}^{p} \bar{r}_{is}^{k}(\phi_s - \pi_s)$] and identical arc-use pseudo costs [i.e., $\sum_{i=1}^{m} a_{ij}(\pi_i - \alpha_i)$], a claim that case 2 exists is simply a claim that the shortest-chain algorithm does not find the shortest chain.

Usefulness of the Procedure

In order to be useful in application, the routes selected and resources assigned must be feasible in the object system. Routes through the network are composed of a series of arcs and the resources assigned to them. Again using a transportation network as an example, it is undesirable to have different vehicle types assigned to contiguous arcs of the same mode in a chain.

That is, one wants the same vehicle to carry the commodity over all contiguous arcs of the same mode in a chain. Quarter-ton and 12-ton trucks may be feasible substitutes, but one does not wish to transfer from one to another at a node.

This is an important consideration if the results of the solution are to be used. It is simply not feasible in practice to use chains that employ different vehicles on various arcs of the same chain unless the chain is multimode and transfer arcs are included. A procedure that is computationally simpler than the general method just described is available, and it guarantees that the same resource combinations will be used on all arcs of a chain that are of a particular mode.

A "master" resource vector representing the resources required to sustain a unit flow over a standard arc of unit length is provided for every commodity, mode, and method. Each arc then has a mode identifier, a condition factor, and a length factor assigned. The minimum-cost (in terms of the simplex multipliers) method is then selected for each iteration, and the resource vectors for each arc are generated using the condition and length scalars. Thus a single master vector, representing a particular method, is selected as the minimum-cost method for all arcs of that mode for each iteration. Continuity of vehicle type is ensured for all chains in the solution.

COMPUTATIONAL EXPERIENCE

A computer program in FORTRAN IV for the Control Data 6400 has been developed for both maximum-flow and minimum-cost formulations incorporating the substitution feature. The program uses the product form of the inverse and will accommodate up to 150 commodities, 550 arcs, and 50 resources. Up to 10 modes are permitted and each mode may have up to three alternative resource-requirement vectors. Thus each arc may use any of three feasible combinations of resources to accomplish the move. A series of test cases has been solved successfully and the results are encouraging with respect to accuracy and speed of solution. The use of the substitution feature does increase the time required for solution, but this increase has been small in the cases tested to date.

REFERENCES

CITED REFERENCES

1. John E. Cremeans and Gene R. Tyndall, "A Model for Optimal Multicommodity Network Flows with Resource Allocation," RAC-P-44, Research Analysis Corporation, Oct 68.
2. L. R. Ford Jr. and D. R. Fulkerson, "A Suggested Computation for Maximal Multi-Commodity Network Flows," Mgt. Sci., Oct 58.
3. J. A. Tomlin, "Minimum-Cost Multi-Commodity Network Flows," Opns. Res., Dec 66.
4. Mandell Bellmore, professor of Operations Research, The Johns Hopkins University. private communications, Mar–Apr 68.
5. Donald D. Boyer, "A Modified Simplex Algorithm for Solving the Multi-Commodity Maximum Flow Problem," TM-14930, The George Washington University Logistics Research Project, Washington, D. C., Mar 69.
6. R. G. D. Allen, Macro-Economic Theory, St. Martin's Press, Inc., New York, 1968, p 36.

ADDITIONAL REFERENCES

Busacker, R. G., et al, "Three General Network Flow Problems and Their Solutions," RAC-SP-183, Research Analysis Corporation, Nov 62.

Dantzig, G. B., and P. Wolfe, "The Decomposition Algorithm for Linear Programming," Econometrica, 29: 767–78 (1961).

Fitzpatrick, G. R., et al, "Programming the Procurement of Airlift and Sealift Forces: A Linear Programming Model for Analysis of the Least Cost Mix of Strategic Deployment Systems," Naval Research Logistics Quart., 14(2): (1967).

Ford, L. R. Jr. and D. R. Fulkerson, Flows in Networks, Princeton University Press, Princeton, N. J., 1962.

Hadley, G., Linear Programming, Addison-Wesley Publishing Co., Inc., Reading, Mass., 1962.

Jewell, William S., "A Primal-Dual Multi-Commodity Flow Algorithm," ORC 66-24, Operations Research Center, University of California, Berkeley, Sep 66.

Rao, M. R., and S. Zionts, "Allocation of Transportation Units to Alternative Trips—A Column Generation Scheme with Out-of-Kilter Subproblems," Opns. Res., Jan–Feb 68.

Sakarovitch, M., "The Multi-Commodity Maximum Flow Problem," Operations Research Center, University of California, Berkeley, Dec 66.

REPRINT OF

DRAFT PAPER, MAY, 1970

THE MULTICOMMODITY NETWORK FLOW MODEL REVISED TO INCLUDE VEHICLES PER TIME PERIOD AND NODE CONSTRAINTS

Enclsoure 3

THE MULTICOMMODITY NETWORK FLOW MODEL REVISED TO INCLUDE VEHICLE PER TIME PERIOD AND NODE CONSTRAINTS

by

Henry S. Weigel
John E. Cremeans

CONTENTS

ABSTRACT

The minimum-cost formulation of the problem of determining multi-commodity flows over a capacitated network subject to resource constraints has been treated in previous papers. In those treatments only capacitated arcs were assumed and a uniform unit of measure like short tons was used for all commodities. This paper treats the effect of constraints on the nodes of the network, allows the commodities to be measured in their "natural" units and allows the network capacities to be expressed in vehicles per time period -- in some cases giving a more accurate representation of the capacities of the network. The solution procedure using the column generation technique is described and computational experience is discussed.

INTRODUCTION

In previous papers[1] the maximum flow multicommodity network problem as formulated by Ford and Fulkerson[2] and the corresponding minimum cost problem as stated by Tomlin[3] were extended to include resource constraints and resource substitution. The purpose of this paper is to describe other modifications to the multicommodity network flow model that have been found to be useful in studies of transportation networks.

These modifications will be described in two steps. First, the problem will be revised to permit the expression of commodity flows in terms of the natural unit of the commodity and the expression of arc capacities in terms of vehicles per time period. In the standard formulation, flows and requirements for commodity movement are converted in into some common unit such as short tons for processing by the algorithm. It is desirable, both for convenience and accuracy, to represent all commodities in terms of their natural units (e.g., numbers of passengers, barrels of gasoline, tons of coal, etc.). It is also convenient to express arc capacities in terms of vehicles per time period since most traffic statistics are collected in this fashion.

Second, node flow constraints will be added to the problem formulation. In the analysis of transportation networks, there are frequently cases where the intersection of three or more arcs has itself a capacity. The capacity of a node is not easily expressed through the addition of dummy arcs. It will be shown that node constraints can be added to permit explicit statement of these capacities.

MINIMUM COST MULTICOMMODITY NETWORK FLOWS WITH RESOURCE CONSTRAINTS

For sake of continuity, the minimum cost problem with resource constraints is restated.

Consider the multimode, multicommodity network $G(N, \mathcal{G})$. N is the set of all nodes of the network. $\mathcal{G}$ is the subset of all ordered pairs (x, y) of the elements of N that are arcs of the network. $\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_m$ is an enumeration of the arcs. Each arc has an associated cost (or distance) $d(x, y) \geq 0$ and each arc has an associated capacity $b(x, y) \geq 0$.

For each commodity $k (k = 1, \ldots, q)$ there is a source s_k and a sink t_k. The flow of the commodity k over arc (x, y) is denoted by $F^k(x, y)$, $k = 1, \ldots, q$. The flows and the arc capacities must be stated in terms of some common unit such as short tons. These flows must satisfy the capacity constraints.

$$\sum_{k=1}^{q} F^k(x, y) \leq b(x, y), \quad (x, y) \in \mathcal{G}$$

In keeping with previous notation, define the set $P^k = \left\{ P_j^{(k)} \middle| P_j^{(k)} \text{ is a chain connecting } s_k \text{ and } t_k \right\}$. Now let P be the union of the sets P^k $(k = 1, \ldots, q)$. Further, let $P_1^{(1)}, P_2^{(1)}, \ldots, P_j^{(k)}, \ldots, P_n^{(q)}$ be the enumeration of the chains $P_j^{(k)} \in P$ such that the subscript j is sufficient to identify the chain, its origin-destination pair, and the commodity with which it is associated.

Thus the k^{th} commodity set is defined by

$$J_k = \left\{ j \middle| P_j^{(k)} \text{ is a chain from } s_k \text{ to } t_k \right\}, \quad k = 1 \ldots, q.$$

The arc-chain incidence matrix is

$$A = [a_{ij}]$$

where

$$a_{ij} = \begin{cases} 1, & \text{if } \mathcal{G}_i \in P_j^{(k)} \\ 0, & \text{otherwise} \end{cases}$$

for $i = 1, \ldots m$; $j = 1, \ldots, n$.

Define the resource matrix for commodity k by

$$R^k = [r_{is}^k] \quad (i = 1, \ldots, m; \; s = 1, \ldots, p)$$

where r_{is}^k is the quantity of resource s required to sustain a unit of

flow of commodity k over arc i; $r_{is}^{k} \geq 0$. And let ρ_s (s = 1, ..., p) be the quantity of resource s available (e.g., in inventory) for assignment to the network.

The objective is to minimize the total cost of the resources used and the tolls on the arcs while meeting delivery requirements and satisfying the arc capacity and resource inventory constraints. Letting $x_j^{(k)}$ (j = 1, ..., n) be the flow of commodity k in chain $P_j^{(k)}$ (j = 1, ..., n; k implicit), b_i the flow capacity of G_i, in arc-chain terms, the minimum cost network flow linear program with resource constraints is:

Minimize $\sum_{j=1}^{n} c_j x_j^{(k)} = z$

subject to

(a) arc capacity constraints

$$\sum_{j=1}^{n} a_{ij} x_j^{(k)} \leq b_i, \quad i = 1, \ldots, m$$

(b) resource constraints

$$\sum_{i=1}^{m} \sum_{k=1}^{q} \sum_{j \in J_k} a_{ij} r_{is}^{k} x_j^{(k)} \leq \rho_s, \quad s = 1, \ldots, p$$

and

(c) delivery requirements

$$\sum_{j \in J_k} x_j^{(k)} = \lambda_k, \quad k = 1, \ldots, q$$

where λ_k is the delivery requirement at t_k, $\lambda_k \geq 0$, (k = 1, ..., q).

The cost coefficient c_j may be defined as:

$$c_j = \sum_{i=1}^{m} \tau_i a_{ij} + \sum_{s=1}^{p} \sum_{i=1}^{m} \Phi_s r_{is}^{k} a_{ij} \quad \text{(for } j = 1, \ldots, n;\ k,$$

where $P_j^{(k)}$ connects s_k and t_k) and

where τ_i is the cost (or toll) for a unit flow over arc i, and Φ_s is the cost of using a unit of resource s.

THE REVISED PROGRAMMING PROBLEM

Use of Natural Units and Vehicles Per Time Period

The capacity $b(x, y)$ of arc $(x, y) \in G$ will now be expressed in vehicles-per-time-period. Some arcs will have vehicle types naturally associated with them. Highway, rail, waterway and airway arcs will be naturally associated with trucks or busses, rail cars, barges and aircraft respectively. For other modes such as pipelines and transfer the "vehicle" must be contrived. One alternative is to use a dummy vehicle which contains exactly one unit of the commodity.

It should be stressed at this point that all commodities will be counted in their natural unit. That is petroleum products will be expressed in barrels, passengers as numbers of passengers and bulk cargo in tons if these are the units best suited to them. Throughout the remainder of this paper each reference to flow or movement of a commodity will be in the natural unit of that commodity.

To express the flow $F^k(x, y)$ in the natural unit of commodity k and the arc constraints in vehicle-per-time-period, re-write the capacity constraints as follows:

$$\sum_{k=1}^{q} \frac{F^k(x, y)}{U^k(x, y)} \leq b(x, y), \quad (x, y) \in G;$$

where $U^k(x, y)$ is the quantity of the commodity k transported over arc (x, y) in a single vehicle, $U^k(x, y) > 0$; and $b(x, y)$ is now in vehicles per time period.

Define a matrix of vehicle per commodity unit coefficients as:

$$V = [v_i^k]$$

where

$$v_i^k = \frac{1}{U^k(x, y)}$$ (the arc (x, y) is denoted as the i^{th} arc by the index i).

In words, v_i^k is the reciprocal of the quantity of commodity k transported over arc i in a single vehicle.

With the above change, the minimum cost linear program can now be stated as follows:

Minimize $$\sum_{j=1}^{n} c_j x_j^{(k)} = z$$

subject to

(a) arc capacity constraints

$$\sum_{j=1}^{n} a_{ij} v_i^k x_j^{(k)} \leq b_i, \quad i = 1, \ldots, m$$

(b) resource constraints

$$\sum_{i=1}^{m} \sum_{k=1}^{q} \sum_{j \in J_k} a_{ij} r_{is}^k x_j^{(k)} \leq \rho_s, \quad s = 1, \ldots, p$$

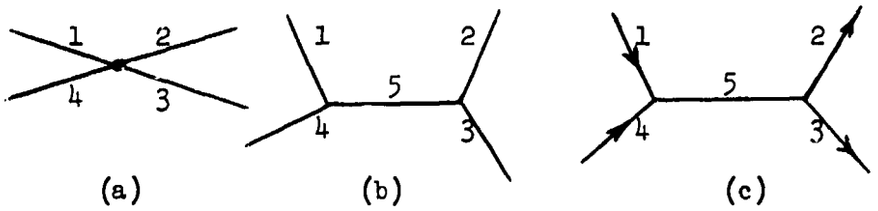
and (c) delivery requirements

$$\sum_{j \in J_k} x_j^{(k)} = \lambda_k, \quad k = 1, \ldots, q.$$

Node Constraints

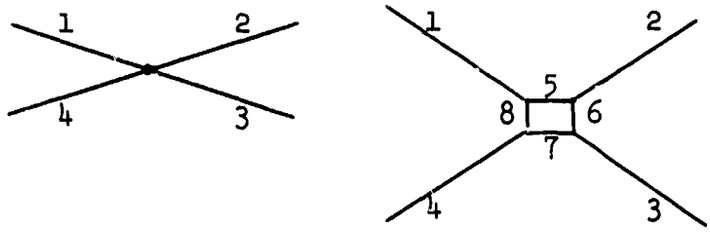
Node constraints allow a direct limitation on the throughput capacity of a node without the necessity of additional assumptions in depicting the node graphically as a small subnetwork. Examples may clarify this point.

In figure 1, the node given in (a) is to be constrained by replacing it with the subnetwork given in (b). The capacity limitation



on arc 5 is that of the node it has replaced. Note, however, that the path 1-4 in (a) contributes to the flow through the node but in (b) it does not contribute to the flow on arc 5. In some cases, this problem may be solved by directing the arcs, i.e., by permitting flow in only one direction as in figure 1-c. This method is not always satisfactory because in a multi-commodity problem the optimum solution may require that one commodity flow over path 1-4 while another is required to flow over path 2-4.

A representation such as that shown in figure 2 is sometimes attempted, but this too is unsatisfactory. What capacity should be assigned to arcs 5, 6, 7, and 8?



(a) (b)

Figure 2

To get an accurate node throughput limitation, a constraint row is added in the problem formulation for each node for which a capacity limitation is desired. Let $\hat{N}$ be the subset of N of those nodes which are to be constrained. To consider the node throughput limitations, let $N_1, \ldots, N_t$ be an enumeration of the nodes $N_\ell \in \hat{N}$. Let $\hat{b}(N_\ell)$ be the capacity associated with the node N which is expressed in vehicles per time period.

Define a matrix of vehicle per commodity unit coefficients for node constraints similar to those for the arc constraints:

$$V = [\hat{v}_\ell^{\,k}]$$

where

$$\hat{v}_\ell^{\,k} = \frac{1}{\hat{U}^k(N_\ell)}$$

and $\hat{U}^k(N_\ell)$ is the quantity of the commodity k transported through node N_ℓ in a single vehicle, $\hat{U}^k(N_\ell) > 0$. Let $\hat{b}_\ell$ be the throughput capacity of node N_ℓ.

The node-chain incidence matrix is $F = [f_{\ell j}]$

where

$$f_{\ell j} = \begin{cases} 1, & \text{if } N_\ell \in P_j^{(k)} \\ 0, & \text{otherwise} \end{cases}$$

for $\ell = 1, \ldots, t$; $j = 1, \ldots, n$.

In arc-chain terms, the minimum cost network flow linear program with resource constraints, node constraints and putting the capacity constraints in vehicles per time period is as follows:

Minimize

$$\sum_{j=1}^{n} c_j x_j^{(k)} = z$$

subject to

(a) arc capacity constraints

$$\sum_{j=1}^{n} a_{ij} v_i^k x_j^{(k)} \leq b_i, \quad i = 1, \ldots, m$$

(b) node capacity constraints

$$\sum_{j=1}^{n} f_{\ell j} \hat{v}_\ell^k x_j^{(k)} \leq \hat{b}_\ell, \quad \ell = 1, \ldots, t$$

(c) resource constraints

$$\sum_{i=1}^{m} \sum_{k=1}^{q} \sum_{j \in J_k} a_{ij} r_{is}^k x_j^{(k)} \leq \rho_s, \quad s = 1, \ldots, p$$

and

(d) delivery requirements

$$\sum_{j \in J_k} x_j^{(k)} = \lambda_k, \quad k = 1, \ldots, q.$$

SOLUTION PROCEDURE USING THE COLUMN GENERATION TECHNIQUE

We have defined a straight forward linear programming problem different from that previously defined only in that node constraints have been added, the natural units are used for the flows and the delivery requirements, and vehicles per time period are used as capacities. It is now necessary to show that the column generation technique can be adapted to these changes. This technique is more

fully described in reference 1.

Define the matrices A*, E, F* and G as follows:

$$A^* = [a^*_{ij}]$$

where

$$a^*_{ij} = a_{ij} v_i^k, \text{ k implied by } P_j^{(k)}.$$

$$E = [e_{sj}]$$

where

$$e_{sj} = \sum_{i=1}^{m} r_{is}^k a_{ij}, \text{ k implied by } P_j^{(k)}.$$

$$F^* = [f^*_{\ell j}]$$

where

$$f^*_{\ell j} = f_{\ell j} \hat{v}_\ell^k, \text{ k implied by } P_j^{(k)}$$

and

$$G = [g_{kj}]$$

where

$$g_{kj} = \begin{cases} 1, \text{ if } j \in J_k \\ 0, \text{ otherwise} \end{cases}$$

$\hat{A}$ is a matrix $(m + p + t + q \times n)$ formed of the submatrices A*, E, F*, and G as follows:

$$\hat{A} = \begin{bmatrix} A^* \\ \hline E \\ \hline F^* \\ \hline G \end{bmatrix}$$

The typical column of $\hat{A}$ is

$$\hat{A}_j = \text{col.}(a^*_{1j}, \ldots, a^*_{mj}, e_{1j}, \ldots, e_{pj}, f^*_{1j}, \ldots, f^*_{tj}, g_{1j}, \ldots, g_{qj}).$$

$\hat{A}$ will be quite large due to the fact that there is typically an enormous number of chains connecting s_k and t_k $(k = 1, \ldots, q)$. The shortest chain algorithm can be used to develop the $\hat{A}_j$ that will satisfy the simplex rule. Further if the shortest chain algorithm can find no chain satisfying the requirement, an optimum has been reached.

This formulation can be solved by adopting the standard two-phased

procedure. Phase I minimizes to zero the value of

$$j = n + m + t + p + q$$
$$\Sigma\, x_j^{(k)}$$
$$j = n + m + t + p + 1$$

to obtain an initial basic feasible solution. This effectively assigns a cost of 1 to the artificial variables and a cost of zero to the other variables in Phase I. Phase II begins with the basic feasible solution determined in Phase I and proceeds to minimize

$$\sum_{j=1}^{n} c_j x_j^{(k)} = z$$

In Phase I, I_{m+p+q} may be used as the initial basis and the simplex rule is to enter a chain in the basis if, and only if,

$$c_j - c_B B^{-1} \hat{A}_j < 0$$

where

$$c_B B^{-1} = (\alpha_1, \ldots, \alpha_m, \pi_1, \ldots, \pi_p, \beta_1, \ldots, \beta_t, \sigma_1, \ldots, \sigma_q)$$

so that the simplex multipliers α_i are associated with the arcs, the π_s are associated with the resources, the β_ℓ are associated with the nodes, and the σ_k are associated with the artificial variables. Thus the vector $\hat{A}_j$ is entered if

$$-\sum_{i=1}^{m} a_{ij} \left[\alpha_i v_i^k + \sum_{s=1}^{p} \pi_s r_{is}^k \right] - \sum_{\ell=1}^{t} f_{\ell j} \hat{v}_\ell^k \beta_\ell < \sigma_k .$$

The contribution of each arc to $c_j - c_B B^{-1} \hat{A}_j$ in Phase I is

$$d_i^k = - \left[\alpha_i v_i^k + \sum_{s=1}^{p} \pi_s r_{is}^k \right]$$; noting that $c_j = 0$ for structural vectors.

The contribution of each node to $c_j - c_B B^{-1} \hat{A}_j$ must be distributed over the arcs to facilitate the use of the shortest chain algorithm. Disallowing the origin and destination nodes from being constrained, it is clear that if a constrained node is contained in a route, the route also contains precisely two of the arcs which emanate from or terminate at this node. This suggests a method for the distribution of the node contributions. Namely, that half of the contribution be added to each arc which has this node as an end point.

To do this, define

$$\delta_{\ell i} = \begin{cases} 1, & \text{if node } N_\ell \text{ is an end point of arc } G_i \\ 0, & \text{otherwise} \end{cases}$$

The effective contribution of each arc to $c_j - c_B B^{-1}\hat{A}_j$ becomes

$$\hat{d}_i^k = -\left[\alpha_i v_i^k + \sum_{s=1}^{p} \pi_s r_{is}^k + \tfrac{1}{2} \sum_{\ell=1}^{t} \delta_{\ell i} \hat{v}_\ell^k \beta_\ell\right]$$

The shortest chain algorithm may be used to find

$$\min_j \left[\sum_{G_i \in P_j(k)} \hat{d}_i^k\right] = \min_j \left[\sum_{G_i \in P_j(k)} \left(-\alpha_i v_i^k - \sum_{s=1}^{p} \pi_s r_{is}^k - \tfrac{1}{2} \sum_{\ell=1}^{t} \delta_{\ell i} \hat{v}_\ell^k \beta_\ell\right)\right]$$

over all k.

Where G_i is the i^{th} arc and $P_j^{(k)}$ is the j^{th} chain from s_k to t_k.

The minimum over all commodities is selected as the candidate to enter the basis. The column vector to leave the basis may be determined in the standard simplex fashion. Should any α_i, π_s or β_ℓ be positive the corresponding slack variable may be entered into the basis.

In Phase II

$$c_j - c_B B^{-1}\hat{A}_j = \sum_{i=1}^{m} \tau_i a_{ij} + \sum_{s=1}^{p} \Phi_s e_{sj} - \sum_{i=1}^{m} v_i^k \alpha_i a_{ij} - \sum_{s=1}^{p} \pi_s e_{sj} - \sum_{\ell=1}^{t} f_{\ell j} \hat{v}_\ell^k \beta_\ell - \sum_{w=1}^{q} \sigma_w g_{wj}$$

$$= \sum_{i=1}^{m} a_{ij}(\tau_i - v_i^k \alpha_i) + \sum_{s=1}^{p} \sum_{i=1}^{m} a_{ij} r_{is}^k (\Phi_s - \pi_s) - \sum_{\ell=1}^{t} f_{\ell j} \hat{v}_\ell^k \beta_\ell - \sum_{w=1}^{q} \sigma_w g_{wj}$$

where

$$e_{sj} = \sum_{i=1}^{m} r_{is}^k a_{ij}$$

and

$$g_{kj} = \begin{cases} 1, & \text{if } j \in J_k \\ 0, & \text{otherwise} \end{cases}$$

Thus

$$d_i^k = \tau_i - v_i^k \alpha_i + \sum_{s=1}^{p} r_{is}^k (\Phi_s - \pi_s)$$

and

$$\hat{d}_i^k = \tau_i - v_i^k \alpha_i + \sum_{s=1}^{p} r_{is}^k (\Phi_s - \pi_s) - \tfrac{1}{2} \sum_{\ell=1}^{t} \delta_{\ell i} \hat{v}_\ell^k \beta_\ell$$

may be assigned to arc i. The shortest, i.e., the chain with the least

$$\sum_{i=1}^{m} \hat{d}_i^k - \sigma_k < 0$$

for k = 1, ..., q, may then be entered in the basis.

Phase II is terminated and the value of z is minimized when, for the minimum j,

$$\sum_{G_i \in P_j(k)} \hat{d}_i^k \geq \sigma_k .$$

MODIFICATION OF THE SUBSTITUTION PROCEDURE

The substitution procedure is affected by the change in that the use of a different group of resources may require that a different coefficient of vehicles per commodity unit be used. Under the procedure described above the matrix V is a two dimensional matrix by arc and by commodity. When substitution is permitted this matrix must be given the additional dimension of substitution group.

Redefine the matrices V and $\hat{V}$, the matrices of vehicles per commodity unit, as:

$$V = [v_{ih}^k] \text{ and } \hat{V} = [\hat{v}_{\ell h}^k] \text{ respectively}$$

where v_{ih}^k is the reciprocal of the quantity of commodity k transported over arc i in a single vehicle determined by substitution group h and $\hat{v}_{\ell h}^k$ is the reciprocal of the quantity of commodity k transported through node ℓ in a single vehicle determined by substitution group h. The coefficients v_{ih}^k and $\hat{v}_{\ell h}^k$ are therefore selected in each iteration based on the minimum "cost" (in terms of simplex multipliers) alternative resource vector to be used.

Summary of the procedure

To summarize, the proposed procedure is:

(1) Calculate $C_B B^{-1} = (\alpha_1, \ldots, \alpha_m, \pi_1, \ldots, \pi_s, \beta_1, \ldots, \beta_t, \sigma_1, \ldots, \sigma_q)$

(2) In the previous formulation, for each arc-commodity pair a single combination of resources is required and represented by the vector,

$$P_i^k = (r_{i1}^k, r_{i2}^k, \ldots, r_{ip}^k), \text{ of the matrix } R^k.$$

Define a new resource matrix: $T = [t_{ik}]$ $(i = 1, \ldots, m; k = 1, \ldots, q)$ where $t_{ik} = \left\{ \hat{R}_i^k \,\middle|\, \hat{R}_i^k \text{ is any feasible resource vector for arc } i, \text{ commodity } k \right\}$ $\hat{R}_i^k = (\hat{r}_{i1}^k, \ldots, \hat{r}_{ip}^k)$.

In words, each element of T is the set of alternative resource vectors for a movement of one unit of commodity k over arc i. For each arc-commodity pair find the least-cost applicable resource vector, "cost" meaning cost in terms of the simplex multipliers and resource prices and call it the vector h.

$$\bar{R}_{ih}^k = \left\{ \hat{R}_i^k \,\middle|\, \sum_{s=1}^{p} \hat{r}_{is}^k (\phi_s - \pi_s) \text{ is minimum for } \hat{R}_i^k \in t_{ik} \right\}$$

where

$$\bar{R}_{ih}^k = (\bar{r}_{i1}^k, \ldots, \bar{r}_{ip}^k)$$

(3) For commodity k = 1, ..., q calculate

$$\hat{d}_i^k = \tau_i - \alpha_i v_{ih}^k + \sum_{s=1}^{p} \bar{r}_{is}^k (\phi_s - \pi_s) - \tfrac{1}{2} \sum_{\ell=1}^{t} \delta_{\ell i} \hat{v}_{\ell h}^k \beta_\ell \text{ for } i = 1, \ldots, m$$

and assign $\hat{d}_i^k$ to arc i as a pseudo cost.

(4) Using the shortest chain algorithm, find the chain with the least

$$\hat{d}_j^k = \sum_{i=1}^{m} a_{ij} \left[\tau_i - \alpha_i v_{ih}^k + \sum_{s=1}^{p} \bar{r}_{is}^k (\phi_s - \pi_s) - \tfrac{1}{2} \sum_{\ell=1}^{t} \delta_{\ell i} \hat{v}_{\ell h}^k \beta_\ell \right] \text{ for } k = 1, \ldots, q.$$

(5) Find $\min_k [\hat{d}_j^k - \sigma_k]$

(6) If the minimum $[\hat{d}_j^k - \sigma_k] < 0$, the vector $\hat{A}_j$ is entered in the basis. If minimum $[\hat{d}_j^k - \sigma_k] \geq 0$ there is no chain that may improve the value of the objective function, and the procedure is terminated.

CONCLUSIONS

It seems to be the consensus of the literature that the basic Ford-Fulkerson multicommodity algorithm attains its speed based in part upon the fact that the conventional method results in a zero-one matrix. It should be emphasized, however, that the literature contains no direct test of this idea. The assumption of speed seems to be based largely on the simplicity of the zero-one matrix and the fact that the decomposition algorithm (that is directly related to the Ford-Fulkerson suggested method) typically does require large amounts of computer time.

The Control Data 6400 computer program for the RACAT algorithm which had been written at the Research Analysis Corporation was revised to include the changes discussed in this paper. It was of interest to make a comparison of the performances of the unrevised and the revised RACAT algorithms. Two problems were used as tests. One, a small problem with 52 arcs, 5 resources, 2 origin-destination pairs and no node constraints, a total of 59 rows. The computer time statistics on the CDC 6400 of two variations of the problem formulation are given in Table 1 below. One formulation has the arc capacities, the movement requirements and the route flows in terms of tons per day. The other has the arc capacities in vehicles per day, the movement requirement and the route flows in the commodities' natural units.

Table 1

COMPUTER TIME STATISTICS FOR PROBLEM 1

Constraint type	Central processor time (sec)	No. of iterations	Time per iteration (sec)
Vehicles per day	13.3	19	.70
Tons per day	11.8	23	.51

In this simple problem one arc constraint type was directly converted from the other to make the problems as much the same as possible. As a result, the objective function values were the same and the flows over respective arcs were the same. Every effort was made to keep the problems similar so that the timing comparisons would be meaningful. The central processor time for the run with the arc constraints in

vehicles per day was 12.7% greater than that for the run with the arc constraints in tons per day.

The second problem which was used as a test did not have the exact conversion from the tons formulation to the vehicles formulation as the first one had.* In the vehicles formulation two arcs of the network were combined into one with the capacity being adjusted accordingly and several resources with identical productivities were combined and the resource inventories were adjusted accordingly. The tons formulation of the problem has 290 arcs, 42 resources, 9 origin destination pairs (2 of them had zero requirements), a total of 341 rows. The vehicles formulation has the same statistics except for one less arc and five less resources. Table 2 gives the run time comparisons on the CDC 6400. As a result of the inexact conversion of the tons formulation to the vehicles formulation, the objective function for the vehicles formulation was slightly smaller, as expected, and the solution was slightly different from the tons formulation. The central processor time for the run with the arc constraints in vehicles per day was 55% greater than that for the run with the arc constraints in tons per day.

Table 2

COMPUTER TIME STATISTICS FOR PROBLEM 2

Constraint type	Central processor time (sec)	No. of iterations	Time per iteration (sec)
Vehicles per day	1860	265	7.0
Tons per day	1200	156	7.7

It has been suggested in the past that the multicommodity network flow problem with resource constraints might best be solved using the decomposition method of Dantzig and Wolfe. With the addition of the V matrix and the destruction of the zero-one nature of the major submatrix, it could now be argued more strongly that decomposition might

*This problem was solved for purposes other than that of testing the algorithm and the time to solve.

be a better approach. It is possible that this is true. No systematic evaluation of the decomposition alternative has been made in the course of the study leading to this paper. It is argued, however, that from a practical standpoint of problem preparation and computer running time the proposed method has distinct advantages over the decomposition method.

The proposed method is a self-contained procedure that will arrive at a solution from the relatively simple inputs of the network, resource productivity data and movement requirements. The decomposition approach would require an elaborate computer program to generate all possible routes using all possible combinations of allowable resources. It is possible, however, that a completely new model using the decomposition principle would result in shorter running times. Budget limitations prohibit the investigation of this possibility at this time.

The revised RACAT algorithm will increase the accuracy in the handling of capacity limitations if it is assumed that a vehicle limitation is a more accurate expression of capacity than a limitation expressed in short tons. Since most capacities expressed in short tons are simple conversions from vehicle per time period figures using the average vehicle load, it seems likely that this is true.

The inclusion of the node constraints allows for a direct limitation on the throughput capacity of a node, rather than the more complex way of depicting a node graphically as a small subnetwork. "Networking" a node is possible only in specialized cases, or additional assumptions like directing the arcs need to be made. These additional assumptions tend to unduly restrict the network.

The revision also permits much more flexible modeling of specialized processes such as transfer, depot and other intra-nodal functions. The capacities of arcs representing such processes do vary depending on the commodity involved. The revised procedure will permit a more realistic handling of these problems.

REFERENCES

CITED REFERENCES

1. Cremeans, J. E. and Tyndall, G. R., "A Model for Optimal Multi-Commodity Network Flows with Resource Allocation," Research Analysis Corp., RAC-P-44, October 1968.

   Cremeans, J. E., Smith, R. A., and Tyndall, G. R., "An Extension to the Multi-Commodity Network Flow Model to Permit Substitution of Resources," Research Analysis Corp., RAC-P-52, April 1969.

2. Ford, L. R. Jr., and Fulkerson, D. R., "A Suggested Computation for Maximal Multi-Commodity Network Flows," Mgt. Sci., Oct 58.

3. Tomlin, J. A., "Minimum-Cost Multi-Commodity Network Flows," Opns. Res. Dec 66.

ADDITIONAL REFERENCES

Boyer, Donald D., "A Modified Simplex Algorithm for Solving the Multi-Commodity Maximum Flow Problem," TM14930, The George Washington University Logistics Research Project, Washington, D. C. Mar 68.

Busacker, R. G., et al, "Three General Network Flow Problems and Their Solutions," RAC-SP-183, Research Analysis Corporation, Nov 62.

Busacker, R. G. and Saaty, T. L., Finite Graphs and Networks, McGraw-Hill Book Company, New York, N. Y., 1965.

Cremeans, J. E., "The Bridges of Konigsburg," ASTME Vectors, May-June, 1969, p. 4.

Charnes, A., "Optimality and Degeneracy in Linear Programming," Econometrica, Vol, 20, April, 1952.

Dreyfus, S. F., "An Appraisal of Some Shortest-Path Algorithms," The RAND Corp., RM-5433-1-PR, Sept 68.

Fitzpatrick, G. R., et al, "Programming the Procurement of Airlift and Sealift Forces: A Linear Programming Model for Analysis of the Least Cost Mix of Strategic Deployment Systems," Naval Research Logistics Quart., 14 (2): (1967).

Ford, L. R. Jr. and Fulkerson, D. R., Flows in Networks, Princeton University Press, Princeton, N. J. 1962.

Gass, S. I., Linear Programming - Methods and Applications, McGraw-Hill Book Co., Inc., New York, 1958.

Hadley, G., Linear Programming, Addison-Wesley Publishing Company, Inc. Reading, Mass. 1962.

Koopmans, T. C., ed., Activity Analysis of Production and Allocation, John Wiley and Son, Inc., New York, N. Y., 1951.

Orchard-Hays, W., Advanced Linear-Programming Computing Techniques, McGraw-Hill Book Co., Inc., New York, 1968.

Rao, M. R., and S. Zionts, "Allocation of Transportation Units to Alternative Trips -- A Column Generation Scheme with Out-of Kilter Subproblems," Opns. Res., Jan-Feb 68.

Sakarovitch, M., "The Multi-Commodity Maximum Flow Problem," Operations Research Center, University of California, Berkeley, Dec 66.

REPRINT OF

RAC STUDY REPORT, MAY 1970

POST OPTIMAL CONSIDERATIONS FOR ETNAM I

RAC Project 809.102
Logistics Department

May 1970

POST OPTIMAL CONSIDERATIONS FOR ETNAM I

J. E. Cremeans
H. S. Weigel

Prepared for the Defense
Communications Agency,
Contract No. DCA 100-70-C-0039

Research Analysis Corporation
McLean, Virginia

This effort was accomplished for the Defense Communications Agency, National Military Command System Technical Support (DCA, NMCSTS) Directorate as part of its overall program of technical support to the J-4 Mobility Plans and Analysis Division, Office of the Joint Chiefs of Staff.

CONTENTS

INTRODUCTION

The purpose of this paper is to study the use of post-optimal procedures in the ETNAM I system of programs. These procedures consist chiefly of two types: 1) parametric linear programming and 2) sensitivity analysis or ranging.

In parametric linear programming, a coefficient (usually selected from the objective function or the right hand side constraints) is treated as a parameter and allowed to vary between specified limits. As the parameter changes, the solution is changed, whenever necessary, to keep it optimal.

Sensitivity analysis is done to find the range of a coefficient for which the current solution remains optimal. This is a very useful tool for analyzing objective function or cost coefficients and right hand side constraint coefficients. For these two types of coefficients this method is known as cost ranging and right hand side ranging respectively.

Currently, the ETNAM optimal solution consists of the objective function, the optimal routes and the amount shipped on each route, the marginal costs or shadow prices and various other specific reports gleaned from the basic optimal solution. Questions of the type "what happens to the solution if I lower my requirements?" or "what happens to the solution if I increase a certain resource inventory?" are usually handled by parametric linear programming in the standard linear programming codes.

In the usual linear programming formulation of the ETNAM-type problem, all possible routes are generated and evaluated by the algorithm.

The essential difference and the great advantage of the ETNAM method is that all possible routes need never be generated. Only those routes that are needed are found by the shortest chain algorithm. Parametric programming is a relatively simple although time consuming procedure in the standard linear program because the possible routes can be evaluated to determine the amount of change in the parameters required to bring about a change in the solution. Thus we can determine the limits of the coefficients by testing the potential routes. The shortest chain algorithm must first be given the values of the coefficients in order to find the best route. What is a workable short-cut in the standard linear program would be a tedious and expensive trial and error procedure for the shortest chain.

In sensitivity analysis, cost ranging also requires an explicit knowledge of all the vectors; but, right hand side ranging appears feasible and may prove to be a useful tool for measuring the sensitivity of the right hand side coefficients.

A mathematical treatment of the post-optimal procedures follows. This treatment assumes a rudimentary knowledge of the RACAT algorithm. The reader should see the references at the end of this paper for background information.

PARAMETRIC LINEAR PROGRAMMING

Consider the following linear prog am in standard form.

$$\min Z = \sum_{j=1}^{n} c_j x_j$$

subject to $\sum_{j=1}^{n} a_{ij}x_j = b_i$, $i = 1, 2, \ldots, m$

$x_j \geq 0$ for $j = 1, 2, \ldots, n$.

or in matrix notation

$$\min Z = CX$$

subject to $AX = b$

$$X \geq 0$$

where, $C = (c_1, c_2, \ldots, c_n)$

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ x_n \end{bmatrix}, \quad b = \begin{bmatrix} b_1 \\ b_2 \\ \cdot \\ \cdot \\ \cdot \\ b_m \end{bmatrix}, \quad A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdot & & & \\ \cdot & & & \\ \cdot & & & \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

and denote a column of A by $A_j = \begin{bmatrix} a_{1j} \\ a_{2j} \\ \cdot \\ \cdot \\ \cdot \\ a_{mj} \end{bmatrix}$

The criterion for optimality is that $c_B B^{-1} A_j - C_j \leq 0$ for all j, or equivalently, $\max_j (c_B B^{-1} A_j - c_j) \leq 0$. The shortest chain algorithm generates the vector with maximum $c_B B^{-1} A_j - c_j$. C_B is the cost coefficient vector of the basic variables and B is an mxm matrix consisting of those A_j which are currently in the basis. Assume for simplicity that the first m variables are in the basis. Then, $C_B = (c_1, c_2, \ldots, c_m)$

$$X_B = \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ x_m \end{bmatrix}, \quad B = (A_1, A_2, \ldots, A_m) = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \cdot & & & \\ \cdot & & & \\ \cdot & & & \\ a_{m1} & a_{m2} & \cdots & a_{mm} \end{bmatrix}$$

Now suppose the first cost coefficient c_1 is to be parametrized between its present value c_1 and some larger value $\bar{c}_1$. Denote the parameters by λ. Then

$$c_1 \leq \lambda \leq \bar{c}_1$$

To determine which vector to enter the basis when λ is increased,

$$C_B B^{-1} A_j - c_j = \lambda h_{1j} + \sum_{i=2}^{m} c_i h_{ij} - c_j,$$

where $\begin{bmatrix} h_{1j} \\ h_{2j} \\ \cdot \\ \cdot \\ \cdot \\ h_{mj} \end{bmatrix} = B^{-1} A_j.$

Increase λ until $\lambda h_{1j} + \sum_{i=2}^{m} c_i h_{ij} - c_j = 0$. Choose the A_j to enter the basis for which

$$\lambda = \min_{j = 1,2,\ldots,n} \left[\frac{c_j - \sum_{i=2}^{m} c_i h_{ij}}{h_{1j}} \;\middle|\; h_{1j} > 0 \right].$$

Hence, an explicit knowledge of all A_j is assumed. As has been noted in the introduction, this is not the case in ETNAM I. Rather, each A_j is generated as it is needed. To generate all A_j would defeat the purpose of the RACAT algorithm.

An alternative to parametric linear programming is to make a change in the data without changing the structure of the problem and "crashing" in the last optimal basis. However, there is no guarantee that this method will save any computer running time. For relatively few and small changes it may, for other changes it may even have an adverse effect.

A similar analysis of the right hand side parametric linear programming case has been made with the same conclusions being drawn.

SENSITIVITY ANALYSIS

In cost ranging, the range of a cost coefficient c_j is determined for which the basis need not be changed; that is, for which the solution remains optimal. The c_j, as used in ETNAM, is a function of the tolls on the arcs, the arcs used in the j^{th} route (A_j), the resource productivities, and the resource prices. So that the range of a resource price or a toll is determined by the range of the c_j's.

Suppose the coefficient to be ranged, c_k, is in the basis. Then the following inequalities determine the range of c_k. $C_B B^{-1} A_j - C_j \leq 0$ for all j such that A_j is not a basic variable. This means that all possible routes, A_j, would have to be known.

If c_k is not in the basis, then $C_B B^{-1} A_k - c_k \leq 0$ determines the range of c_k. But, since c_k is not in the basis, A_k has not yet been generated and A_k is an unknown making it impossible to solve this inequality for c_k.

Since not all the A_j are known in the ETNAM formulation, cost ranging cannot currently be done. However, right hand side ranging can be done.

To find the range of b_k, the following set of inequalities must be solved.

$B^{-1} b \geq 0$ or writing them out,

$$\beta_{11} b_1 + \ldots + \beta_{1k} b_k + \ldots + \beta_{1m} b_m \geq 0$$

$$\beta_{21} b_1 + \ldots + \beta_{2k} b_k + \ldots + \beta_{2m} b_m \geq 0$$

$$\vdots$$

$$\beta_{m1} b_1 + \ldots + \beta_{mk} b_k + \ldots + \beta_{mm} b_m \geq 0$$

where

$$B^{-1} = \begin{bmatrix} \beta_{11} & \beta_{12} & \cdots & \beta_{1m} \\ \beta_{21} & \beta_{22} & \cdots & \beta_{2m} \\ \vdots & & & \\ \beta_{m1} & \beta_{m2} & \cdots & \beta_{mm} \end{bmatrix}$$

The i^{th} inequality can be re-written as

$$b_k \geq \frac{-(\beta_{i1} b_1 + \cdots + \beta_{ik-1} b_{k-1} + \beta_{ik+1} b_{k+1} + \cdots + \beta_{im} b_m)}{\beta_{ik}} \quad \text{for } \beta_{ik} > 0$$

and

$$b_k \leq \frac{-(\beta_{i1} b_1 + \cdots + \beta_{ik-1} b_{k-1} + \beta_{ik+1} b_{k+1} + \cdots + \beta_{im} b_m)}{\beta_{ik}} \quad \text{for } \beta_{ik} < 0$$

Denote the upper and lower bounds of b_k, by $\bar{b}_k$ and $\underline{b}_k$ respectively.

Then

$$\underline{b}_k = \max_i \left[\frac{-(\beta_{i1} b_1 + \beta_{ik-1} b_{k-1} + \beta_{ik+1} b_{k+1} + \cdots \beta_{im} b_m)}{\beta_{ik}} \;\middle|\; \beta_{ik} > 0 \right]$$

$$\bar{b}_k = \min_i \left[\frac{-(\beta_{i1} b_1 + \cdots + \beta_{ik-1} b_{k-1} + \beta_{ik+1} b_{k+1} \cdots + \beta_{im} b_m)}{\beta_{ik}} \;\middle|\; \beta_{ik} < 0 \right]$$

The inequlaity for which $\beta_{ik} = 0$ contributes nothing to the upper or lower bound of the range of b_k.

It has been shown here that right hand side ranging can be done in ETNAM I.

CONCLUSIONS AND RECOMMENDATIONS

It has been shown in the above analysis that it is possible to do right hand side ranging in the ETNAM model. But within the present framework of ETNAM I, parametric linear programming and cost ranging cannot be done. Based upon this analysis it is feasible to implement right hand side ranging in the ETNAM system of computer programs. This

would allow the user to analyze the sensitivity of arc capacities, resource inventory constraints, requirements and any other constraint that may be added at some future date.

The programming task does not appear too great since it is possible to write the right hand side ranging routine as a separate program module with only minor changes, if any, to the other three program modules of ETNAM. Virtually all of the available computer core is being used in the RACAT and OUTPUT modules. Incorporating this post optimal procedure in either of these modules would necessitate a reallocation and economization of computer core. This would also make the debugging more difficult.

It is therefore recommended that a separate ETNAM program module be written to do right hand side ranging. Its use would be optional to the user.

REFERENCES

Cremeans, J. E. and Tyndall, G. R., "A Model for Optimal Multi-Commodity Network Flows with Resource Allocation," Research Analysis Corp., RAC-P-44, October 1968.

Cremeans, J. E., Smith, R. A., and Tyndall, G. R., "An Extension to the Multi-Commodity Network Flow Model to Permit Substitution of Resources," Research Analysis Corp., RAC-P-52, April 1969.

Cremeans, J. E. and Weigel, H. S., System Description for ETNAM I: PART I, Users Manual, Oct 1969, PART II, Analytical Manual, Oct 1969.

Gass, S. I., Linear Programming - Methods and Applications, McGraw-Hill Book Co., Inc., New York, 1958.

Hadley, G., Linear Programming, Addison-Wesley Publishing Company, Inc. Reading, Mass. 1962.

SAMPLE LISTINGS OF DATA FOR INPUT MODULE

1. Sample Problem. The first listing is the input data of the actual reports of the test problem shown in the SD, Volume I, User's Manual. There are no substitution, 4 modes, 2 commodities, 5 resources, 2 node constraints. Link capacities are expressed in vehicles-per-time-period.

2. Partial Listing. The second listing omits the complete network. This sample shows 2 substitutions, 4 modes, 2 commodities, 7 resources, no node constraints. Link capacities are expressed as vehicles-per-time period. The productivity cards PROD and PALD are listed showing the handling of substitution of resources. The payload factor cards, TRIP, also show the handling of substitution of resources.

3. Partial Listing. The last listing also omits the complete network. This sample shows no substitution of resources, 4 modes, 2 commodities, 5 resources, no node constraints. Link capacities are expressed as tons-per-time-period. The conversion factor card CONF is shown. The origin-destination nodes ODPR have open delivery requirement so the requirements by commodity cards DELI are shown. There is also a sample TOLL card.

```
...ETNAM I SAMPLE PROBLEM ... SEPTEMBER 1970
1                15   20   25   30   35   40   45   50
↓                ↓    ↓    ↓    ↓    ↓    ↓    ↓    ↓
MIN COST         1    4    2    5    2    1   75  150

1         10    16    22    28    34
↓         ↓     ↓     ↓     ↓     ↓
NAMO      HIWY  RAIL  TRAN  PORT
NACO      DRY   PAX
NARE      S/PL  BUS   FRTC  PAXC  TSC
ARCS
1       7          18    24      32      40          50 51
↓       ↓          ↓     ↓       ↓       ↓           ↓↓
01H     05H        1     1     125     700           111
01H     11H        1     1      90     700           111
02H     03H        1     1     120     900           111
02H     06H        1     1     190     200           111
03H     04H        1     1      90     800           111
03H     06H        1     1     140     700           111
04H     06H        1     1     190     700           111
05H     06H        1     1     250     300           111
05H     02H        1     1     160     400           111
05H     11H        1     1     105     400           111
07H     08H        1     1      60     700           111
07H     11H        1     1      85     700           111
08H     09H        1     1     165     400           111
08H     02H        1     1     190     400           111
09H     04H        1     1     220     400           111
10H     04H        1     1     375      50           111
11H     02H        1     1     125     400           111
11H     09H        1     1     150     500           111
01R     11R        2     1      95     700           111
01R     12R        2     1     220     700           111
02R     03R        2     1     135     700           111
02R     06R        2     1     215      30           111
03R     04R        2     1      95     700           111
04R     06R        2     1     215     700           111
07R     13R        2     1      70     700           111
08R     02R        2     1     200     700           111
08R     10R        2     1     110     700           111
09R     04R        2     1     250     700           111
10R     04R        2     1     410      50           111
11R     02R        2     1     140     700           111
12R     03R        2     1     170     700           111
13R     02R        2     1     140     500           111
13R     09R        2     1     115     590           111
07R     08R        2     1      65     700           111
01H     01R        3     1       1       1           111
02H     02R        3     1       1       1           111
03H     03R        3     1       1       1           111
04H     04R        3     1       1       1           111
06H     06R        3     1       1       1           111
07H     07R        3     1       1       1           111
08H     08R        3     1       1       1           111
09H     09R        3     1       1       1           111
```

```
10H   10R          3      1       1       1          111
04H   55D          3      1       1       1         1101
04H   65B          3      1       1       1         1011
06H   55D          3      1       1       1         1101
06H   65D          3      1       1       1         1011
50S   01H          4      1       1       1         1101
50S   07H          4      1       1       1         1101
60S   07H          4      1       1       1         1011
60S   08H          4      1       1       1         1011
60S   10H          4      1       1       1         1011
 (BLANK CARD)
1          10         20
↓          ↓          ↓
03R        2          500
11R        2          180
1          10    15   20        30              40   45   50        60
↓          ↓     ↓    ↓         ↓               ↓    ↓    ↓         ↓
INVE                  1      100000                    2    100000
INVE                  3      100000                    4    100000
INVE                  5      100000
PHIS                  1           7                    2         7
PHIS                  3           1                    4         1
PHIS                  5           2
RATE                  1          15                    2        12
RATE                  3          .1                    4        .2
PROD       1     1    1        1E-3              1    2    3     104E-5
PROD       1     3    5        3E-5              1    4    5     139E-5
PROD       2     1    2       25E-5              2    2    4    8286E-8
PROD       2     4    5      143E-8
TRIP       1     1    1          10              1    1    2        10
TRIP       1     1    3        7000              1    1    4     10000
TRIP       1     2    3       14000              1    2    4      7000
TRIP       1     2    1          40              1    2    2        50
1          10    14   20                  35    40   44   50              65
↓          ↓     ↓    ↓                   ↓     ↓    ↓    ↓               ↓
ODPR
ODPR       1     50S  55D          9080       2    60S  65D         13475
ODPR     999
END
```

THERE IS ONLY ONE BLANK CARD IN THE DATA, OTHERS ARE FOR CLARITY
CARD COLUMNS ARE INDICATED BY ARROWS WITH COLUMN NUMBERS ABOVE THEM

```
.ETNAM I SAMPLE LISTING WITH 2 SUBST, NO NODE CONSTRAINTS, VEHICLES/TIME PERIOD
1                15   20   25   30   35   40   45   50
MIN COST         2    4    2    7    0    1    75   150
1          10   16   22   28   34   40   46
NAMO       HIWY RAIL TRAN PORT
NACO       DRY  PAX
NARE       S/PL BUS  FRTC PAXC TSC  SCAR CCAR
ARCS
1    7          18   24     32     40       50 51
01H  05H        1    1      125    700        111
01H  11H        1    1      90     700        111
02H  03H        1    1      120    900        111
02H  06H        1    1      190    200        111
03H  04H        1    1      90     800        111
          .
          .
          .
60S  07H        4    1      1      1          101
60S  08H        4    1      1      1          101
60S  10H        4    1      1      1          101
 (BLANK CARD)
1          10   15   20       30     35   40   45   50       60     65
INVE                1    100000                   2    100000
INVE                3    2000                     4    50
INVE                5    100000                   6    100000
INVE                7    100000
PHIS                1    7                        2    7
PHIS                3    1                        4    1
PHIS                5    2                        6    1
PHIS                7    1
RATE                1    15                       2    12
RATE                3    .1                       4    .2
PROD       1    1    1    1E-3        1    2    3    104E-5
PROD       1    1    51   1E-3        1    2    57   104E-4
PROD       2    1    2    25E-5       2    2    6    8286E-7
PROD       2    1    52   25E-5       2    2    54   8286E-8
PROD       2    4    5    143E-8      2    4    55   143E-8
PALD       1    3    5    33333  1    1    4    5    719    1
PALD       1    3    55   33333  1    1    4    55   719    1
TRIP       1    1    1    10          1    1    2    10
TRIP       1    1    3    7000        1    1    4    10000
TRIP       1    2    3    14000       1    2    4    7000
TRIP       1    2    1    40          1    2    2    30
TRIP       2    1    2    *           2    2    2    50
1          10   14   20             35   40   44   50             65
ODPR
ODPR       1    50S  55D       9080     2    60S  65B       13475
ODPR  999
END
```

THERE IS ONLY ONE BLANK CARD IN THE DATA, OTHERS ARE FOR CLARITY
CARD COLUMNS ARE INDICATED BY ARROWS WITH COLUMN NUMBERS ABOVE THEM

```
.ETNAM I SAMPLE LISTING WITH 1 SUBST, NO NODE CONSTRAINTS, TONS
1               15    20    25    30    35    40    45    50
↓               ↓     ↓     ↓     ↓     ↓     ↓     ↓     ↓
MIN COST        1     4     2     5     0     0     75    150
1         10    16      22      28      34
↓         ↓     ↓       ↓       ↓       ↓
NAMO      HIWY  RAIL    TRAN    PORT
NACO      DRY   PAX
NARE      S/PL  BUS     FRTC    PAXC    TSC
ARCS
1       7             18      24        32        40          50 51
↓       ↓             ↓       ↓         ↓         ↓           ↓↓
01H     05H           1       1         125       7000          111
01H     11H           1       1          90       7000          111
02H     03H           1       1         120       9000          111
02H     06H           1       1         190       2000          111
03H     04H           1       1          90       6000          111
              .
              .
              .
60S     07H           4       1          1       10000          101
60S     08H           4       1          1       10000          101
60S     10H           4       1          1       10000          101
 (BLANK CARD)
1         10    15    20          30          40    45    50          60
↓         ↓     ↓     ↓           ↓           ↓     ↓     ↓           ↓
INVE                  1       100000                      2       100000
INVE                  3       100000                      4       100000
INVE                  5       100000
PHIS                  1            7                      2            7
PHIS                  3            1                      4            1
PHIS                  5            2
RATE                  1           15                      2           12
RATE                  3           .1                      4           .2
PROD      1     1     1         1E-3          1     2     3       104E-5
PROD      1     3     5         3E-5          1     4     5       139E-5
PROD      2     1     2        25E-5          2     2     4      8286E-8
PROD      2     4     5       143E-8
CONF                  2            5
DELI                  1         9080                      2        13475
1         10    14    20                35    40    44    50
↓         ↓     ↓     ↓                 ↓     ↓     ↓     ↓
ODPR
ODPR      1     50S   03H                       1     50S   12R
ODPR      2     60S   06H                       2     60S   03H
ODPR    999
TOLL
TOLL      1     03H   04H         100000
TOLL    999
END
```

THERE IS ONLY ONE BLANK CARD IN THE DATA, OTHERS ARE FOR CLARITY
CARD COLUMNS ARE INDICATED BY ARROWS WITH COLUMN NUMBERS ABOVE THEM

SELECTED BIBLIOGRAPHY

General Background Information

Busacker, R. G. and Saaty, T. L., Finite Graphs and Networks, McGraw-Hill Book Company, New York, N. Y. 1965.

Cremeans, J. E., "The Bridges of Konigsburg," ASTME Vectors, May-June, 1969, p. 4.

Hadley, G., Linear Programming, Addison-Wesley Publishing Company, Inc., Reading Mass., 1962.

Ford, L. R. Jr and Fulkerson, D. R., Flows in Networks, Princeton University Press, Princeton, N. J. 1962.

Orchard-Hays W., Advanced Linear Programming Computing Techniques, McGraw-Hill Book Co., Inc., New York, 1968.

Theoretical Papers

Cremeans, J. E. and Tyndall, G. R., "A Model for Optimal Multi-Commodity Network Flows with Resource Allocation," Research Analysis Corp., RAC-P-44, October 1968.

Cremeans, J. E., Smith, R. A., and Tyndall, G. R., "An Extension to the Multi-Commodity Network Flow Model to Permit Substitution of Resources," Research Analysis Corp., RAC-P-52, April 1969.

Cremeans, J. E., Smith, R. A., and Tyndall, G. R., "Optimal Multicommodity Network Flows with Resource Allocation," Naval Research Logistics Quart., 17(3): Sept. 1970.

Dreyfus, S. F., "An Appraisal of Some Shortest Path Algorithms," The RAND Corp., RM-5433-1-PR, Sept. 68.

Ford, L. R. Jr and Fulkerson, D. R., "A Suggested Computation for Maximal Multi-Commodity Network Flows," Mgt. Sci., Oct 58.

Tomlin, J. A., "Minimum-Cost Multi-Commodity Network Flows," Opns. Res., Dec 66.

Additional References

Boyer, Donald D., "A Modified Simplex Algorithm for Solving the Multi-Commodity Maximum Flow Problem," TM-14930, The George Washington University Logistics Research Project, Washington, D. C., Mar 68.

Busacker, R. G., et al, "Three General Network Flow Problems and Their Solutions," RAC-SP-183, Research Analysis Corporation, Nov 62.

Charnes, A., "Optimality and Degeneracy in Linear Programming," Econometrica, Vol 20, April, 1952.

Fitzpatrick, G. R., et al. "Programming the Porcurement of Airlift and Sealift Forces: A Linear Programming Model for Analysis of the Least Cost Mix of Strategic Deployment Systems," Naval Research Logistics Quart., 14 (2): (1967).

Enclosure 6

Preceding page blank

Additional References

Koopmans, T. C., ed., Activity Analysis of Production and Allocation, John Wiley and Son, Inc., New York, N. Y. 1951.

Rao, M. R. and S. Zionts, "Allocation of Transportation Units to Alternative Trips -- A Column Generation Scheme with Out-of-Kilter Subproblems," Opns. Res., Jan-Feb 68.

Sakarovitch, M., "The Multi-Commodity Maximum Flow Problem," Operations Research Center, University of California, Berkeley, Dec 66.